HAL
open science

Bases of relations in one or several variables: fast algorithms and applications

Vincent Neiger

To cite this version:

Vincent Neiger. Bases of relations in one or several variables: fast algorithms and applications. Symbolic Computation [cs.SC]. École Normale Supérieure de Lyon - University of Waterloo, 2016. English. NNT : 2016LYSEN052 . tel-01431413

HAL Id: tel-01431413

https://theses.hal.science/tel-01431413v1

Submitted on 10 Jan 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

École Doctorale en Informatique et Mathématiques de Lyon (ED512)

THÈSE

pour l'obtention du

Doctorat de l'Université de Lyon

(spécialité informatique)

délivré par

l'École Normale Supérieure de Lyon

présentée et soutenue publiquement le 30 novembre 2016

par

Vincent Neiger

Bases of relations in one or several variables: fast algorithms and applications

---

Bases de relations en une ou plusieurs variables : algorithmes rapides et applications

Composition du jury

Rapporteurs :	Bernhard Beckermann	Université de Lille
	Mark van Hoeij	Florida State University
Examinateurs :	Jean-Charles Faugère	Inria, Université Pierre et Marie Curie, Paris VI
	Nadia Heninger	University of Pennsylvania
	Marie-Françoise Roy	Université de Rennes 1
Directeurs de thèse:	Claude-Pierre Jeannerod	Inria, École Normale Supérieure de Lyon
	Éric Schost	University of Waterloo
	Gilles Villard	CNRS, École Normale Supérieure de Lyon

Laboratoire de l'Informatique du Parallélisme — UMR 5668

Remerciements

Un grand merci à Bernhard Beckermann et Mark Van Hoeij pour avoir accepté de rapporter cette thèse, et à Jean-Charles Faugère, Nadia Heninger et Marie-Françoise Roy pour avoir accepté de faire partie du jury. Je vous suis extrêmement reconnaissant pour le temps que vous accordez à ce travail et pour l'intérêt que vous lui portez.

Un merci tout aussi grand à Claude-Pierre, Éric et Gilles, pour avoir supervisé et encadré mes travaux pendant ces trois ans ainsi qu'en stage auparavant. Travailler avec vous a été un réel plaisir, et je vous remercie d'avoir été généreusement attentifs au bon déroulement de ma thèse, sur tous les plans. J'aurais de grandes difficultés à émettre la moindre critique négative sur votre encadrement, indépendamment du fait que l'endroit serait tout à fait inapproprié.

Merci aux nombreuses personnes avec qui j'ai pu avoir des discussions lors de diverses conférences ou déplacements. Flûte, voici le passage redouté des remerciements où l'on scrute cette zone de la mémoire où sont enfouies certaines rencontres qui nous ont marqués et où surnagent d'autres souvenirs sans intérêt particulier. Pas de doute que je vais oublier des noms, alors pour les gens concernés : désolé.

Merci à Bruno, Pascal, Romain ; à Olivier ; à Laurent ; à Clément ; à Armin, Esmaeil, Javad, Muhammad ; à Albert, Andrew, Arne, George, Seny, Steve, Suzy ; à Guénaël, Jérémy, Ludovic, Mohab ; à Guillaume, Hugo, Pierre-Jean, Svyat ; à Daniel, Johan, Luca ; à Thomas S.-P., Louis ; et à toute l'équipe à Lyon.

Merci aux collègues thésards Adeline, Antoine, Fabrice, Philippe, Matei, Sébastien M., Sébastien T., Silviu, Thomas P., Valentina. Merci à Chiraz, Damien et Évelyne pour leur travail efficace et souriant. Merci à Christophe pour avoir encadré mon stage de M1, qui m'a donné un début de goût pour la recherche.

Merci enfin à Sandra, Robert, Julia et Thomas, pour votre gentillesse, votre bonne humeur, votre ouverture d'esprit. Je garde un bon souvenir des séjours dans votre famille.

Pendant les deux premières années, ce doctorat était réalisé en cotutelle avec Western University (London, Canada), où travaillait alors Éric Schost. À ce titre, mon travail a été soutenu financièrement par les bourses de mobilité du *Programme Avenir Lyon Saint-Étienne*, de la *Région Rhône-Alpes* (CMIRA - Explo'ra Doc), ainsi que par *Mitacs* (Globalink Research Award - Inria).

‘Pour tout bagage on a vingt ans
on a l’expérience des parents. . . ’

À mes parents,

Pour m’avoir pondu, élevé, éduqué,
puis supporté tant d’années,
en toutes circonstances.
✍

À mes frères,

Pour toutes les choses partagées,
rigolades, disputes, sport,
repas, musique, etc.
☕

À Clémentine,

Pour tes rires, ta douceur, tes mots,
et tous les instants de bonheur,
passés et surtout à venir
🐈

‘. . . pour qu’éclatent de joie
chaque heure et chaque jour.’

Contents

List of Problems xi

List of Algorithms xiii

List of Tables xv

List of Figures xv

Preamble 1

Part I Problems and overview of contributions 15

Chapter 1 Generating sets of modules over polynomial rings 19

1.1 Popov bases of modules over univariate polynomial rings 20

1.1.1 Bases and polynomial matrices 20

1.1.2 Row degrees and shifted reduced forms 22

1.1.3 Pivots and shifted Popov forms 27

1.2 Designing fast algorithms for shifted Popov bases 30

1.2.1 Finding and using the minimal degree 30

1.2.2 Size of bases and target costs 35

1.3 Gröbner bases of modules over multivariate polynomial rings 38

1.3.1 Generating sets of ideals and modules 39

1.3.2 Monomial orders and initial module 41

1.3.3 Gröbner bases 46

1.3.4 Link with shifted Popov bases 48

1.3.5 Modules of finite (co)dimension and multiplication matrices . . 50

Chapter 2 Fast computation of relation bases **55**

2.1 Relations or syzygies in finite-dimensional modules 55

2.1.1 Gröbner bases of multivariate modules of relations 55

2.1.2 Univariate case: minimal relation bases 60

2.1.3 Overview of our results 62

2.2 Fast algorithms for dense multiplication matrices 63

2.2.1 Results 63

2.2.2 Overview of our algorithm 65

2.2.3 Change of monomial order for zero-dimensional ideals 68

2.3 Multiplication matrix in nilpotent Jordan form 70

2.3.1 Link with minimal approximant bases 70

2.3.2 Overview of previous work 71

2.3.3 Computing shifted Popov approximant bases 72

2.4 Multiplication matrix in Jordan form 75

2.4.1 Link with minimal interpolant bases 75

2.4.2 Algorithm for almost uniform shifts 76

2.4.3 Computing shifted Popov interpolant bases 78

2.5 Companion-block diagonal multiplication matrix 80

2.5.1 Link with systems of linear modular equations 81

2.5.2 Computing shifted Popov solution bases 82

2.5.3 Computing a solution via structured linear algebra 86

Chapter 3 Impact on related problems **89**

3.1 Multivariate interpolation and list-decoding algorithms 89

3.1.1 Multivariate interpolant with degree constraints 89

3.1.2 List-decoding of (folded) Reed-Solomon codes 92

3.1.3 Computing shifted Popov bases of multivariate interpolants . . 96

3.1.4 Soft-decoding of Reed-Solomon codes 100

3.1.5 General Coppersmith technique over $\mathbb{K}[X]$ 101

3.2 Computing shifted Popov forms of polynomial matrices 105

3.2.1 Overview 105

3.2.2 Computing shifted Popov forms for arbitrary shifts 109

3.2.3 Deterministic computation of Hermite forms and determinants . 110

Part II Relation bases for arbitrary multiplication matrices 115

Chapter 4 Computing relation bases via linear algebra 119

4.1 The linear algebra viewpoint 120

4.1.1 Linearization: viewing polynomial relations as scalar relations . 120

4.1.2 Bounded-degree relations and nullspace of multi-Krylov matrices 122

4.1.3 Multi-Krylov matrices in the univariate case 125

4.2 Fast computation of the monomial basis 126

4.2.1 Row rank profile and monomial basis 127

4.2.2 Structure and row rank profile of a multi-Krylov matrix 128

4.2.3 Computing the row rank profile of a multi-Krylov matrix 129

4.3 Fast computation of the relation basis 134

4.3.1 Simultaneous computation of normal forms of monomials 134

4.3.2 Univariate case: computing shifted Popov relation bases 136

4.3.3 Computing reduced Gröbner relation bases 138

Chapter 5 Computing multiplication matrices from a Gröbner basis 141

5.1 Structural properties of the monomial basis 141

5.2 The case of two variables 142

5.3 Computing rows of a Krylov matrix 144

5.4 Computing the multiplication matrices 144

Part III Systems of linear modular univariate equations 151

Chapter 6 Preliminaries and ingredients 155

6.1 Multiplication time functions for polynomials and polynomial matrices 155

6.2 Using the minimal degree to ensure uniform shift and output degrees . 156

6.3 Computing residuals for systems of linear modular equations 158

6.4 Iterative relation basis for a triangular multiplication matrix [BL00] . . 160

Chapter 7 Computing shifted Popov approximant bases 165

7.1 Fast algorithms for almost uniform orders [GJV03] 166

7.2 Arbitrary orders: reduction to the case $n \in \mathcal{O}(m)$ 172

7.3 Fast approximant bases in Popov form with known minimal degree . . 175

7.4 Fast approximant bases in Popov form for arbitrary shifts 181

Chapter 8 Computing shifted Popov solution bases **185**
8.1 Fast algorithm via kernel bases when the minimal degree is known . . . 186
8.2 The case of one equation . . . 190
8.2.1 Amplitude, splitting indices, and block triangular shape . . . 192
8.2.2 Fast algorithm for a single equation . . . 194
8.3 Fast solution bases in Popov form for arbitrary shifts . . . 198

Chapter 9 Computing a solution via structured linear algebra **201**
9.1 Solving structured homogeneous linear systems . . . 201
9.2 Reducing to solving a mosaic-Hankel linear system . . . 204
9.3 Directly computing a solution via a Toeplitz-like system . . . 208

Chapter 10 Coppersmith technique over the univariate polynomials **215**
10.1 The approach based on row reduction . . . 215
10.2 Reducing to a system of linear modular equations . . . 218
10.2.1 Introduction: the specific case $d = 1$ . . . 218
10.2.2 The general case $d \geqslant 1$ . . . 220

Part IV Interpolant bases and multivariate interpolation **225**

Chapter 11 Multivariate interpolation and list-decoding **229**
11.1 Reducing Problem 11 to Problem 10 . . . 229
11.2 Impact on decoding algorithms for Reed-Solomon codes . . . 234
11.2.1 Interpolation step of the Guruswami-Sudan algorithm . . . 234
11.2.2 Re-encoding technique . . . 235
11.2.3 Interpolation step in the Wu algorithm . . . 237
11.2.4 Slowdown due to repeating points in the soft-decoding . . . 239
11.3 The approach based on row reduction . . . 240
11.4 On assumption $\mathcal{H}_{\text{int},1}$ . . . 242
11.5 On assumption $\mathcal{H}_{\text{int},3}$ . . . 243

Chapter 12 Some tools for computing with polynomial matrices **245**
12.1 More time functions for polynomial matrices . . . 245
12.2 Multiplying matrices with unbalanced row degrees [ZLS12] . . . 248
12.3 Detailed cost bound for the kernel basis algorithm of [ZLS12] . . . 249

Chapter 13 Computing shifted Popov interpolant bases **255**
13.1 Divide-and-conquer approach for a triangular multiplication matrix . . 255
13.2 Fast interpolant bases in reduced form for almost uniform shifts 257
13.3 Fast interpolant bases in Popov form for arbitrary shifts 261

Chapter 14 Details of new ingredients for interpolant bases **265**
14.1 Fast shifted reduction of a reduced matrix 265
14.2 Computing residuals for interpolant bases 267
14.2.1 Residuals and Chinese remaindering 268
14.2.2 Main algorithm 270
14.2.3 Computing the residual by shifting $\mathbf{P}$ 272
14.2.4 Computing the residual by Chinese remaindering 274
14.3 Computing interpolant bases with known minimal degree 277

Part V Normal forms of polynomial matrices **281**

Chapter 15 Shifted Popov forms **285**
15.1 The generic determinant degree bound 285
15.2 Reducing to almost uniform input degrees 286
15.2.1 Column partial linearization 287
15.2.2 Row partial linearization 291
15.2.3 Reducing the degrees in shifted Popov form computation 292
15.3 Fast, probabilistic computation of the shifted Popov form 293

Chapter 16 Hermite form and determinant **297**
16.1 Preliminaries: column bases 298
16.2 Computing the diagonal entries of a triangular form 299
16.2.1 Fast block elimination 300
16.2.2 Computational cost and example 300
16.3 Fast computation of the determinant of a polynomial matrix 303
16.4 Fast Hermite form algorithm with known minimal degree 309
16.4.1 Hermite form via shifted column reduction 309
16.4.2 Reducing the amplitude of the minimal degree 310
16.4.3 Algorithm and computational cost 314
16.4.4 Proof of Lemma 16.19 317

16.5 Reduction to almost uniform input degrees 319

Perspectives **321**

Index **325**

Bibliography **329**

List of Problems

1 *Gröbner basis of a syzygy module* 56
2 *Gröbner relation basis* 58
3 *Gröbner basis of a syzygy module defined by a dual basis* 59
4 *Minimal relation basis* 62
5 *Change of monomial order* for a zero-dimensional ideal 68
6 *Computing the multiplication matrices* from a Gröbner basis 69
7 *Minimal approximant basis* 71
8 *Minimal interpolant basis* 76
9 *Minimal solution basis* 81
10 *Small solution vector* 86

11 *Constrained multivariate interpolation* 90
12 *Minimal basis of multivariate interpolants* 98
13 *Small modular roots* (Coppersmith technique over $\mathbb{K}[X]$) 102
14 *Interpolation step* of the Coppersmith technique over $\mathbb{K}[X]$ 103
15 *Shifted Popov form* of a square nonsingular polynomial matrix 106
16 *Hermite form* of a square nonsingular polynomial matrix 106

17 *Change of basis* in the interpolation step of Coppersmith technique 220

List of Algorithms

1 KrylovRankProf: Row rank profile of a multi-Krylov matrix 131
2 LinNormalForm: Normal forms via linear algebra 135
3 LinPopovRelBas: Shifted Popov relation bases via linear algebra 137
4 LinRelBas: Reduced Gröbner relation bases via linear algebra 139

5 KrylovEval: Computing rows of a Krylov matrix 145
6 NextExpSet: Computing the next exponent set $\mathcal{S}_i$ 147
7 MulMat: Multiplication matrices from reduced Gröbner basis 149

8 IterRelBas: Iterative relation basis for triangular mult. mat. 161

9 LinAppBas: Approximant basis with identical orders: base case [GJV03] . 167
10 IterAppBas: Iterative approximant basis with identical orders [GJV03] . 169
11 DaCAppBas: Divide-and-conquer app. basis with identical orders [GJV03] 170
12 ReduceNbEqAppBas: Approximant basis: reduction to $n \in \mathcal{O}(m)$ 173
13 MinDegAppBas: Approximant basis with known minimal degree 179
14 FastPopovAppBas: Shifted Popov approximant basis 182

15 PivDegKerBas: Shifted Popov kernel basis with known pivot degree . . 190
16 MinDegSolBas: Solution basis with known minimal degree 191
17 FastPopovSolBasOneEq: Solution basis for a single equation 197
18 FastPopovSolBas: Shifted Popov solution basis 199

19 SolVecViaHankel: Small solution vector via a mosaic-Hankel system . . 209
20 SolVecViaToeplitz: Small solution vector via a Toeplitz-like system . . 212

21 ChangeBasisOneStep: Change of basis in Coppersmith: from P to YP 222
22 ChangeBasis: Change of basis in Coppersmith technique 223

23 MultiIntViaSolVec: Finding multivariate interpolants as solution vectors 233
24 GurSudIntReenc: Guruswami-Sudan interpolation step with re-encoding 237

25 RDegPolMatMul: Multiplication with unbalanced row degrees [ZLS12] 250
26 MinKerBas: Shifted minimal kernel basis [ZLS12] 252

27 DaCRelBas: Divide-and-conquer relation basis for triangular mult. mat. 256
28 MinIntBas: Minimal interpolant basis 258

29 ShiftMinIntBas: Shifted minimal interpolant basis 260
30 FastPopovIntBas: Shifted Popov interpolant basis 262

31 ChangeShift: Shifted reduced form of a reduced matrix 267
32 JordanMul: Residuals for a multiplication matrix in Jordan form 271
33 JordanMulByShifting: Jordan residual via shifting 273
34 JordanMulByCRT: Jordan residual via Chinese remaindering 275
35 MinDegIntBas: Interpolant basis with known minimal degree 278

36 SPopovForm: Shifted Popov form of a polynomial matrix 295

37 HermiteDiag: Diagonal entries of the Hermite form 301
38 Determinant: Determinant of a nonsingular polynomial matrix 306
39 MinDegHermite: Hermite form with known minimal degree 315

List of Tables

1 *Improvement:* Gröbner basis of relation module 7
2 *Improvement:* computing the multiplication matrices 7
3 *Improvement:* systems of linear modular equations (known roots) 9
4 *Improvement:* approximant bases 9
5 *Improvement:* systems of linear modular equations (arbitrary moduli) . . . 10
6 *Improvement:* Reed-Solomon list-decoding ($\mu = \mu_1 = \cdots = \mu_\nu$, distinct x_i's) 11
7 *Improvement:* folded Reed-Solomon list-decoding 11
8 *Improvement:* Shifted Popov and Hermite forms of a matrix 12

2.1 Fast algorithms for systems of linear modular equations over $\mathbb{K}[X]$ 83

3.1 Fast algorithms for the interpolation step of Guruswami-Sudan list-decoding 94
3.2 Fast algorithms for shifted reduced forms and shifted Popov forms of a polynomial matrix 108

List of Figures

1.1 Staircase of the bivariate monomial ideals $\langle X^8, Y\rangle$ and $\langle X^8, X^2Y^4, Y^7\rangle$ . . 42
1.2 Staircase of a Borel-fixed monomial ideal in three variables 42

10.1 Shape of the known basis of the module in Copersmith technique 216

Preamble

In this document, we aim at designing fast algorithms to solve problems from computer algebra and connected fields such as coding theory. Our results improve upon those known previously for a number of situations. This includes the faster computation of canonical solution bases for systems of linear modular equations over the univariate polynomials, and of normal forms of polynomial matrices. We exploit this to accelerate the interpolation step of the Guruswami-Sudan and Kötter-Vardy algorithms for the list- and soft-decodings of Reed-Solomon codes. In addition, we give a faster algorithm for the change of monomial order for zero-dimensional multivariate ideals whose initial ideal is Borel-fixed.

Our main problem asks to compute relations between elements in a vector space, which is finite-dimensional and equipped with multiplication matrices that make it a module over a multivariate polynomial ring. More specifically, it asks to compute the Gröbner basis of the syzygy module defined by these elements, for a given monomial order. Many of our algorithms focus on modules over the univariate polynomials; then, the syzygy module is free, monomial orders are specified by shifts, and Gröbner bases are known as shifted Popov bases. This general setting encompasses fundamental algorithmic questions about univariate polynomials, such as Padé approximation and rational interpolation, and from linear algebra, such as finding the minimal polynomial of a vector.

Our results were obtained by elaborating over ideas and techniques developed for such questions, with in particular Keller-Gehrig's algorithm for computing the characteristic polynomial of a matrix [KG85] and Beckermann-Labahn's divide-and-conquer computation of minimal approximant bases [BL94]; the latter follows on from work on the half-gcd algorithm [Knu70, Sch71, Moe73] and Padé approximation [BGY80, CC86]. We both extend these approaches to more general contexts and introduce new ingredients for faster algorithms, in particular through a better control of the degrees during the computation.

⋆⋆⋆

Cost model

Our goal is to design *fast* sequential algorithms. To measure this quality, it is customary in computer algebra to employ the notion of *arithmetic cost*, that we describe now.

In all the problems we will study, we work over some field $\mathbb{K}$. The efficiency of our algorithms is measured by a cost bound, which provides an upper bound on the number of basic operations in $\mathbb{K}$ performed by the algorithm to compute its output. This bound is usually expressed in terms of parameters that describe the size of the input and of the output, where by size we mean the number of field elements that are used to represent these objects. The basic operations in $\mathbb{K}$ are additions, multiplications, opposites, inverses,

and equality testing, which are all supposed to have unit cost.

Our cost bounds will always be asymptotic estimates with constant factors hidden via the notation $\mathcal{O}(\cdot)$, and sometimes polylogarithmic terms omitted via the notation $\tilde{\mathcal{O}}(\cdot)$. In most cases we will not have any requirement regarding the base field $\mathbb{K}$. There are two exceptions where we require that the cardinality of $\mathbb{K}$ be large enough: first, for problems from interpolation or from coding theory which involve a certain number of distinct points, and second to ensure a good probability of success in the case of probabilistic algorithms.

⋆ ⋆ ⋆

Univariate polynomials and rational reconstruction

One of the important problems in computer algebra and coding theory is the reconstruction of rational functions: given a modulus $\mathfrak{m} \in \mathbb{K}[X]$ of degree $D > 0$ and a polynomial $f \in \mathbb{K}[X]$ of degree less than D, it asks to compute polynomials $p_1, p_2 \in \mathbb{K}[X]$ such that

$$p_1 f = p_2 \bmod \mathfrak{m}, \quad \deg(p_1) < N_1, \quad \text{and} \quad \deg(p_2) < N_2 \tag{1}$$

for some prescribed bounds $N_1, N_2 \in \mathbb{Z}_{>0}$. These are often such that $N_1 + N_2 = D + 1$, thus ensuring the existence of a solution which may be found by linear system solving, as the coefficients of p_1 and p_2 form a vector in the left nullspace of a $(D+1) \times D$ matrix.

Eq. (1) is a single linear modular equation with two unknowns: one of the main goals in this document is to design fast algorithms for systems of linear modular equations with several unknowns. In particular, we will borrow from previous work on solving Eq. (1), which we outline now; we refer to [GG13, Sec. 5.7 to 5.9] for a more detailed overview.

Depending on the moduli, we consider three cases of Eq. (1), which will all play a key role in our algorithms for generalizations of this reconstruction problem.

The case $\mathfrak{m} = X^D$, over the field of complex numbers, is known in analysis as *Padé approximation* [Cau21, Her93, Pad94]. Early algorithms used recursion formulas in the Padé table to find an approximant in $\mathcal{O}(D^2)$ floating-point operations; this is discussed for example in [Bak75, Cla75, BGM96]. Padé approximation was later considered from a symbolic computation viewpoint, starting with [Ged73, Ged79].

In the meantime, a solution was independently proposed for solving the *key equation* in the context of decoding BCH codes [Ber68, Ch. 7], which also corresponds to $\mathfrak{m} = X^D$. The algorithm was then modified in [Mas69] into what is known as the Berlekamp-Massey algorithm; it uses $\mathcal{O}(D^2)$ operations in $\mathbb{K}$. The fact that this also computes the minimal generator of a linearly recurrent sequence over $\mathbb{K}$ was highlighted in [Zie68, Mas69, Mil75].

The case of a modulus which splits, $\mathfrak{m} = \prod_{1 \leqslant k \leqslant D}(X - x_k)$ for some known field elements $(x_1, \ldots, x_D)$, is called multipoint Padé approximation [Bak75, Ch. 8] and Hermite rational interpolation [War74, War76], or Cauchy interpolation [Cau21] [GG13, Sec. 5.8] when the roots are distinct. The latter situation occurs in the decoding of Reed-Solomon codes via the Welch-Berlekamp approach [WB86].

In relation with the decoding of Goppa codes, Eq. (1) was considered in its general form in [SKHN75], where it was solved via the Euclidean algorithm in $\mathcal{O}(D^2)$ operations; this approach can also be found in [MS78] or [GG13, Sec. 5.7]. The intimate connections between the algorithms of Euclid, of Berlekamp-Massey, and for Padé approximants have been extensively studied [Mil75, WS79, BGY80, Che84, CC86, Dor87].

In a nutshell, and from a computer algebra point of view, there are two approaches for fast algorithms. The extended GCD-based approach works in a top-down fashion by first considering the high-degree coefficients of the input polynomials. Relying on a divide-and-conquer scheme similar to the half-gcd algorithm of Knuth-Schönhage-Moenck [Knu70, Sch71, Moe73], a fast variant of this approach was obtained in [GY79, BGY80]. The Padé-based approach rather operates in a bottom-up manner, starting with the coefficients of low degree, with divide-and-conquer variants given in [BGY80, CC86].

In both cases, the divide-and-conquer algorithms can be roughly sketched as follows. First, they do not compute a single solution (p_1, p_2) but a 2×2 matrix over $\mathbb{K}[X]$. After splitting the input in two parts of half the degree, they recursively solve the problem for the first part, update the second part accordingly, recursively solve the problem for this updated part, and finally combine both recursive solutions via multiplication of 2×2 polynomial matrices. The updated part is called the *residual*.

This yields a cost bound of $\mathcal{O}(\mathsf{M}(D)\log(D))$ operations in $\mathbb{K}$, where the function $\mathsf{M}(\cdot)$ is such that univariate polynomials of degree at most D over $\mathbb{K}$ can be multiplied using $\mathsf{M}(D)$ operations in $\mathbb{K}$. It follows from [Sch77, Nus80] that $\mathsf{M}(D)$ can be taken in $\mathcal{O}(D\log(D)\log(\log(D))) \subset \mathcal{O}\tilde{}(D)$.

⋆⋆⋆

A linear algebra point of view

We now discuss how the above reconstruction problem can be interpreted as the search of a linear relation between the rows of a matrix over $\mathbb{K}$. While this linear algebra viewpoint is unnecessary in the design of fast algorithms for rational reconstruction as in Eq. (1), it will be crucial concerning generalizations to systems of linear modular equations.

To see why, note that both the GCD- and Padé-based algorithms solve Eq. (1), which involves two unknowns in $\mathbb{K}[X]$, by considering 2×2 polynomial matrices in the process. For systems with m unknowns, the algorithms will manipulate $m \times m$ polynomial matrices. One may expect that, at some point such as in the base case of a recursion, these matrices will have low degree or even be constant, and then the problem will be close to computing nullspaces of matrices over $\mathbb{K}$. Therefore, it becomes important to rely not only on fast polynomial multiplication, but also on fast tools from $\mathbb{K}$-linear algebra.

The set $\mathcal{M} = \mathbb{K}[X]/\langle \mathfrak{m} \rangle$ is a $\mathbb{K}$-vector space with basis $\{1, X, \ldots, X^{D-1}\}$. Representing an element $f \in \mathcal{M}$ by its coefficient vector $\mathbf{f} \in \mathbb{K}^{1\times D}$, the multiplication by X in $\mathcal{M}$ is given by a *multiplication matrix* $\mathbf{M} \in \mathbb{K}^{D\times D}$: the coefficients of $Xf \bmod \mathfrak{m}$ are $\mathbf{fM}$.

In the case of Padé approximation ($\mathfrak{m} = X^D$), $\mathbf{M}$ is the upper shift matrix

$$\mathbf{M} = \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & 0 & 1 \\ & & & 0 \end{bmatrix}.$$

More generally, for an arbitrary modulus $\mathfrak{m}$, the multiplication matrix is the companion

matrix built from the coefficients $\mathfrak{m} = c_0 + c_1 X + \cdots + c_{D-1} X^{D-1} + X^D$:

$$\mathbf{M} = \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & 0 & 1 \\ -c_0 & -c_1 & \cdots & -c_{D-1} \end{bmatrix}.$$

When $\mathfrak{m} = \prod_{1 \leqslant k \leqslant D} (X - x_k)$ with known roots, we will also use another basis for $\mathcal{M}$. For example, if the roots are distinct then we can use the Lagrange basis. In this case $f \in \mathcal{M}$ is represented by its evaluation vector $\mathbf{f} = [f(x_1) \ \cdots \ f(x_D)]$, and since the evaluations of Xf are $[x_1 f(x_1) \ \cdots \ x_D f(x_D)]$, the multiplication matrix is the diagonal

$$\mathbf{M} = \begin{bmatrix} x_1 & & & \\ & x_2 & & \\ & & \ddots & \\ & & & x_D \end{bmatrix}.$$

Let us go back to the reconstruction problem, with degree constraints $N_1 = D+1$ and $N_2 = 0$ to simplify matters: we look for p of degree at most D such that $pf = 0 \bmod \mathfrak{m}$. Since the modular product $pf \bmod \mathfrak{m}$ is represented by $\mathbf{f}p(\mathbf{M})$, computing a vector in the left nullspace of the Krylov matrix

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{fM} \\ \vdots \\ \mathbf{fM}^D \end{bmatrix} \in \mathbb{K}^{(D+1) \times D}$$

yields the coefficient vector of a solution polynomial p. For an arbitrary matrix $\mathbf{M}$, the set $\{p \in \mathbb{K}[X] \mid \mathbf{f}p(\mathbf{M}) = 0\}$ is an ideal of $\mathbb{K}[X]$, whose monic generator is the minimal polynomial of the vector $\mathbf{f}$ and is a factor of the characteristic polynomial of $\mathbf{M}$.

The coefficient vector of this generator is the unique nullspace element whose rightmost nonzero entry has smallest index and is 1, and can be computed in $\mathcal{O}(D^\omega \log(D))$ operations in $\mathbb{K}$, as follows. Here, ω is so that we can multiply $m \times m$ matrices over $\mathbb{K}$ in $\mathcal{O}(m^\omega)$ operations in $\mathbb{K}$, the best known bound being $\omega < 2.38$ [CW90, LG14]. First, the Krylov matrix is formed within the above bound by repeated squaring of $\mathbf{M}$, as in [KG85, Sec. 3]. Then, we conclude by Gaussian elimination in $\mathcal{O}(D^\omega)$ operations [IMH82]. For early numerical algorithms along the same lines, we refer to [Kry31, Dan37].

This linear algebra approach will provide an efficient base case for systems of modular equations, thanks to its generality: it works for any $\mathbf{M} \in \mathbb{K}^{D \times D}$ and directly extends to $\mathbf{F} \in \mathbb{K}^{m \times D}$ having rows $\mathbf{f}_1, \ldots, \mathbf{f}_m$. In this case, we consider the $\mathbb{K}[X]$-module

$$\{(p_1, \ldots, p_m) \in \mathbb{K}[X]^m \mid \mathbf{f}_1 p_1(\mathbf{M}) + \cdots + \mathbf{f}_m p_m(\mathbf{M}) = 0\},$$

whose elements such that $\deg(p_i) \leqslant D$ form linear relations between the rows of

$$\begin{bmatrix} \mathbf{F} \\ \mathbf{FM} \\ \vdots \\ \mathbf{FM}^D \end{bmatrix} \in \mathbb{K}^{m(D+1) \times D}.$$

An overview of results related to this situation is given in [Kai80, Chap. 6]. In particular, this module admits canonical bases, called Popov and Hermite bases. These are $m \times m$ matrices over $\mathbb{K}[X]$, with the former having uniformly small degrees and the latter being triangular; they correspond to specific sets of vectors in the nullspace of the above matrix.

As an interesting particular case, the identity matrix $\mathbf{F} = \mathbf{I} \in \mathbb{K}^{D \times D}$ leads to the Popov basis $X\mathbf{I} - \mathbf{M}$, whose determinant is the characteristic polynomial of $\mathbf{M}$. Furthermore, the Hermite basis is known to have the same determinant, and its triangular form allows us to retrieve it efficiently. The characteristic polynomial algorithm of Keller-Gehrig [KG85, Sec. 5] can be seen as the computation of the Hermite basis in $\mathcal{O}(D^\omega \log(D))$ operations.

For systems of linear modular equations, where $\mathbf{F}$ may be any matrix in $\mathbb{K}^{m \times D}$, ideas from [Kai80, KG85] and subsequent research lead to a similarly fast computation of Popov or Hermite bases. One of our contributions is to extend this approach so as to work in multivariate contexts and to cover a wide range of output canonical bases.

⋆⋆⋆

Some ingredients for fast algorithms

Two common threads in our fast algorithms for systems of linear modular equations are the use of divide-and-conquer schemes adapted from Beckermann-Labahn's algorithm [BL94] and, for the base case of the recursion, of ideas from Keller-Gehrig's algorithm.

The algorithm of [BL94] applies the bottom-up approach to the more general Hermite-Padé approximation. In this context, one is given a vector $\mathbf{f} \in \mathbb{K}[X]^{m \times 1}$ and a tuple $\mathbf{N}$ of m degree constraints, and the equation is $\mathbf{pf} = 0 \bmod X^D$ for some unknown $\mathbf{p} \in \mathbb{K}[X]^{1 \times m}$. Eq. (1) corresponds to $m = 2$ and $\mathbf{f}$ being the transpose of $[f \quad -1]$.

Then, it computes an $m \times m$ matrix over $\mathbb{K}[X]$ whose rows form a basis of the set of all solutions without degree constraints. To ensure that one of these rows satisfies them nevertheless, the algorithm computes specific bases which are said to be $\mathbf{s}$-minimal, where the *shift* $\mathbf{s}$ is a tuple of m integers that are related to the constraints in $\mathbf{N}$.

In [BL94], the base case is for $D = 1$ and is solved by Gaussian elimination. Then, when $D \geqslant 2$, the following recursion is used:

1. Compute an $\mathbf{s}$-minimal basis $\mathbf{P}^{(1)}$ of approximants for $\mathbf{f}$ modulo $X^{D/2}$;
2. Compute the residual $\mathbf{g}$ from $\mathbf{P}^{(1)}$ and $\mathbf{f}$, and update $\mathbf{s}$ into $\mathbf{t}$;
3. Compute a $\mathbf{t}$-minimal basis $\mathbf{P}^{(2)}$ of approximants for $\mathbf{g}$ modulo $X^{D/2}$;
4. Return the product $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$.

After the first recursive call, which has processed the low-degree part of $\mathbf{f}$, the residual is an update of the high-degree part which then allows us to continue the computation. Formally, $\mathbf{g} = (X^{-D/2}\mathbf{P}^{(1)}\mathbf{f}) \bmod X^{D/2}$. At the same time, the shift $\mathbf{t}$ corresponds to an update of the degree constraints, depending on the degrees in $\mathbf{P}^{(1)}$.

For Hermite-Padé approximation and some generalizations, this provides a solution in $\tilde{\mathcal{O}}(m^\omega D)$ field operations, which is similar to the cost bound $\mathcal{O}(m^\omega \mathsf{M}(D))$ for multiplying two $m \times m$ matrices with entries of degree at most D [CK91]. One main obstacle towards faster algorithms is the non-uniformity and unpredictability of the degrees in the bases.

Our first objective is to solve systems of equations efficiently for shifts $\mathbf{s}$ whose entries are uniform or close to it: in this case, the average degree of the *rows* in the output basis is

in $\mathcal{O}(D/m)$. Concerning Hermite-Padé approximation, this was exploited in [Sto06, ZL12] to reduce to a case with well controlled degrees solved in [GJV03], leading to the cost bound $\tilde{\mathcal{O}}(m^{\omega-1}D)$. This result and the ideas in these references can be used in some of the more general situations we tackle. In situations where they seem not to apply, we rather control the degrees by introducing a change of shift at each node of the recursion described above, ensuring that only uniform shifts are dealt with in recursive calls.

A second goal is to support arbitrary shifts and to compute $\mathbf{s}$-Popov bases, which are canonical forms of the $\mathbf{s}$-minimal bases. While they have the advantage of having average *column* degree at most D/m independently of $\mathbf{s}$, they do not behave well regarding the product $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$. To solve this, we adapt the divide-and-conquer scheme: we use the bases computed recursively to find degree information on the sought basis, and then exploit this information to reduce to an instance with an almost uniform shift. Instead of multiplying bases together, we solve this instance efficiently as sketched in the previous paragraph.

Another issue with the above divide-and-conquer scheme is that the recursive tree has D leaves and each of them uses $\mathcal{O}(m^2)$ operations. This makes a total cost of $\mathcal{O}(m^2D)$ operations, which is beyond our target cost bound $\tilde{\mathcal{O}}(m^{\omega-1}D)$. We solve this by choosing $D \approx m$ as the base case; in this context, we thus aim at the cost bound $\tilde{\mathcal{O}}(D^\omega)$. Besides, the average column degree of the $\mathbf{s}$-Popov basis is at most $D/m \approx 1$, suggesting that fast polynomial matrix multiplication should not be essential for a fast solution. We obtain the desired efficiency via the linear algebra approach based on Keller-Gehrig's algorithm.

⋆ ⋆ ⋆

Some notations and conventions

A list of the recurring notations in this document can be found at the end of the index.

We will often manipulate tuples of integers, that is, elements of $\mathbb{Z}^m$ for some $m \in \mathbb{Z}_{>0}$. Unless indicated otherwise, all tuple operations and comparisons are componentwise. The set of matrices of dimensions $m \times n$ over a ring $\mathcal{R}$ is denoted by $\mathcal{R}^{m\times n}$. For simplicity of presentation, in the cost analyses we only consider the case $\omega > 2$. All our algorithms still work if $\omega = 2$, with cost bounds that are similar to the ones we give, except that they may involve additional logarithmic factors. Hereafter, $\log(\cdot)$ stands for the logarithm in base 2, which we stress by writing $\log_2(\cdot)$ when any ambiguity should be avoided.

Most our improvements consist in providing algorithms that have the best known cost bound or extend the scope of the fastest known algorithms, for example by using less restrictive assumptions. In a few cases, our contribution is to give a deterministic algorithm with a cost bound that had previously only been achieved by probabilistic algorithms. In what follows, these are always of the Las Vegas type [Pap94, Chap. 11].

⋆ ⋆ ⋆

Multivariate relations and change of monomial order

Generalizing the linear algebra approach above to the multivariate case, we consider a finite-dimensional $\mathbb{K}[X_1, \ldots, X_r]$-module, given to us as $\mathcal{M} = \mathbb{K}^D$ along with a multiplication matrix $\mathbf{M}_i \in \mathbb{K}^{D\times D}$ for each variable; we write $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$. Then, for some elements $\mathbf{f}_1, \ldots, \mathbf{f}_m$ in $\mathcal{M}$, we are interested in the module of relations

$$\mathrm{Syz}_{\mathbf{M}}(\mathbf{F}) = \{(p_1, \ldots, p_m) \in \mathbb{K}[X_1, \ldots, X_r]^m \mid \mathbf{f}_1 p_1(\mathbf{M}) + \cdots + \mathbf{f}_m p_m(\mathbf{M}) = 0\},$$

known as a syzygy module. Given the matrices, the module elements, and a monomial order $\prec$ on $\mathbb{K}[X_1, \ldots, X_r]^m$, our problem is to find the $\prec$-reduced Gröbner basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

Improvement: Gröbner basis of relation module

[FGLM93]	$\mathcal{O}(rD^3)$	general case
[FGHR14]	$\mathcal{O}(D^\omega \log(D) + r\mathsf{M}(D)\log(D))$	$\prec_{\mathrm{lex}}$, Shape Position
Here (Theorem 2.13)	$\mathcal{O}(rD^\omega \log(D))$	general case

As sketched above in the univariate case, the sought Gröbner basis corresponds to a set of vectors in the nullspace of a multi-Krylov matrix. Our algorithm essentially generalizes Keller-Gehrig's approach to the multivariate context and to arbitrary monomial orders.

As summarized below, in addition to computing the multiplication matrices from a description of $\mathcal{M}$, the FGLM algorithm [FGLM93] solves this problem. It is written for ideals ($m = 1$) in [FGLM93], but the ideas extend to the case of modules [Fit97]. Roughly, our improvement comes from an efficient grouping of the operations, thus performing few matrix-matrix products, as opposed to many matrix-vector products in [FGLM93].

In a specific context related to polynomial system solving, Keller-Gehrig's approach had already been used similarly in [FGHR14, Sec. 3], under the restrictive assumption that the ideal is in Shape Position. Focusing on the lexicographic order, this reference gives a probabilistic algorithm in $\mathcal{O}(D^\omega \log(D) + r\mathsf{M}(D)\log(D))$; a deterministic algorithm is also proposed, yet it has the same cost bound as ours while requiring extra assumptions.

In the problem of the change of monomial order, we are given a $\prec_1$-Gröbner basis of a zero-dimensional ideal $\mathcal{I}$ of degree D, as well as another order $\prec_2$, and we want to compute the $\prec_2$-Gröbner basis of $\mathcal{I}$. The efficient algorithms in [FGLM93, FGHR14, FM17] can be seen as a two-step strategy, where one first computes the multiplication matrices $\mathbf{M}$, and then computes the $\prec_2$-Gröbner basis of some module of relations as above.

More precisely, $\mathbb{K}[X_1, \ldots, X_r]/\mathcal{I}$ is isomorphic to $\mathcal{M} = \mathbb{K}^D$ via its monomial basis; then, $\mathcal{I} = \mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ for $\mathbf{F} = [1 \quad 0 \quad \cdots \quad 0] \in \mathbb{K}^{1\times D}$, which stands for the monomial 1. Thus, we have seen that the second step can be done in $\mathcal{O}(rD^\omega \log(D))$ operations. Under the assumption that the $\prec_1$-initial ideal of $\mathcal{I}$ is Borel-fixed, we obtain a similar cost bound for the first step.

Improvement: computing the multiplication matrices

[FGLM93]	$\mathcal{O}(rD^3)$	general case
[FGHR14]	0	$\prec_{\mathrm{drl}}$, $\mathbf{M}_r$ only
Here (Theorem 2.14)	$\mathcal{O}(rD^\omega \log(D))$	Borel-fixed initial ideal

If $\mathcal{I}$ is in Shape Position, then one only needs to compute $\mathbf{M}_r$. This case is studied in [FGHR14], where $\prec_1$ is the degree-reverse lexicographic order $\prec_{\mathrm{drl}}$. This reference shows that $\mathbf{M}_r$ can be directly read off from the input $\prec_{\mathrm{drl}}$-Gröbner basis when $\mathcal{I}$ is a generic ideal, or after a random linear change of coordinates. On the other hand, FGLM computes all matrices with a cubic cost in general: $\prec_1$, $\prec_2$, and $\mathcal{I}$ are arbitrary [FGLM93].

Our result lies in the middle, with a better cost than [FGLM93] and a more general situation than [FGHR14]. Precisely, we only assume that the $\prec_1$-initial ideal of $\mathcal{I}$ is Borel-fixed, which holds after a random linear change of coordinates if $\prec_1$ refines the degree. Then, elaborating over the work in [FGHR14], we compute all the multiplication matrices in $\mathcal{O}(rD^\omega \log(D))$ field operations.

⋆ ⋆ ⋆

Systems of linear modular equations with known roots

Now, we consider systems of linear modular equations over $\mathbb{K}[X]$ of the form

$$\begin{cases} p_1 f_{11} + \cdots + p_m f_{m1} & = & 0 \mod \mathfrak{m}_1 \\ \vdots & & \vdots \\ p_1 f_{1n} + \cdots + p_m f_{mn} & = & 0 \mod \mathfrak{m}_n \end{cases} \tag{2}$$

where each modulus is known through its roots and multiplicities. In a matrix form, this system can be written as $\mathbf{pf} = 0 \bmod \mathfrak{M}$, where $\mathbf{f} = [f_{ij}]_{i,j}$, $\mathfrak{M} = \mathrm{diag}(\mathfrak{m}_1, \ldots, \mathfrak{m}_n)$, and the unknown is $\mathbf{p} = [p_i]_i$.

As hinted at above, bases of solutions to such a system have cardinality m and can be represented as $m \times m$ matrices over $\mathbb{K}[X]$. Given a shift $\mathbf{s} \in \mathbb{Z}^m$, which stands for a monomial order and allows us to take possible degree constraints into account, we will compute bases of solutions that are $\mathbf{s}$-minimal. More precisely, our goal is to compute the canonical one called the $\mathbf{s}$-Popov basis.

Working over $\mathbb{K}[X]$, the notions of $\prec$-reduced Gröbner bases and of $\mathbf{s}$-Popov bases are equivalent, so that our problem here is a particular case of the computation of Gröbner bases for modules of relations $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$. Here, the multiplication matrix $\mathbf{M} \in \mathbb{K}^{D \times D}$ is defined by the moduli, $\mathbf{F} \in \mathbb{K}^{m \times D}$ is deduced from $[f_{ij}]_{i,j}$, and $D = \deg(\mathfrak{m}_1) + \cdots + \deg(\mathfrak{m}_n)$.

In [Bec92, VBB92], an $\mathbf{s}$-minimal basis is computed in $\mathcal{O}(m^2 D^2)$ operations, by an iterative algorithm which processes the roots one after another. It admits a divide-and-conquer variant, extending the approach of [BL94] outlined above to this case.

However, in this situation where roots may be repeated and shared by several moduli, computing the residual is an obstacle. Here, we solve this by mixing polynomial matrix multiplication with Chinese remaindering techniques. The latter allows us to convert between representations of the polynomials f_{ij} by their coefficients and by their evaluations, which we alternatively use depending on the repetition of the roots. With this ingredient, the divide-and-conquer version computes an $\mathbf{s}$-minimal basis in $\tilde{\mathcal{O}}(m^\omega D)$ operations.

Previously, cost bounds quasi-linear in D had only been obtained in the particular case where all moduli have the form $\mathfrak{m}_i = X^{D/n}$, known as *approximant basis* computation since it generalizes Padé approximation. The cost $\tilde{\mathcal{O}}(m^{\omega-1} D)$ was achieved by exploiting the facts that the output bases have degree at most D/n [GJV03], and furthermore that they have small *average* row degree assuming that the entries of $\mathbf{s}$ are small [Sto06, ZL12].

Here, in our more general context, the second bound on the average row degree holds when $\mathbf{s}$ is small, giving hope for an algorithm in $\tilde{\mathcal{O}}(m^{\omega-1} D)$. However, the first bound D/n does not hold, and thus we cannot rely on the same ideas as in [Sto06, ZL12]. The main difficulty is that our assumption that $\mathbf{s}$ has small entries is not preserved in recursive

calls. To overcome this, we first give an algorithm which works only for the the uniform shift $\mathbf{s} = (0, \ldots, 0)$. We have seen that in the divide-and-conquer approach, the shift is modified during the computation: we show how to rely on a fast change of shift to ensure that all recursive calls are with the uniform shift. Using the linear algebra point of view at the base case of the recursion, this achieves the cost bound $\tilde{\mathcal{O}}(m^{\omega-1}D)$.

Now, without requiring assumptions on the shift, the $\mathbf{s}$-Popov basis has small average column degrees. In particular, using normalization at each step of the above-mentioned iterative algorithm, it uses only $\mathcal{O}(mD^2)$ operations [BL00] (see Section 6.4 for the cost analysis). However, the shifted Popov form does not behave well with respect to multiplication, breaking the divide-and-conquer scheme of [BL94].

Here, we adapt this recursion so as to compute the shifted Popov basis in $\tilde{\mathcal{O}}(m^{\omega-1}D)$, for an arbitrary shift. Instead of multiplying the bases obtained recursively, we only use them to deduce the degree shape of the sought basis. Once this degree information is found, we manage to reduce the problem to a shift which has small entries, and thus can be handled by our first algorithm.

Improvement: systems of linear modular equations (known roots)

Previous fastest [BL00]	$\mathcal{O}(mD^2)$	shifted Popov basis
Here (Theorem 2.19)	$\mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(D)^2)$	minimal basis, small shift
Here (Theorem 2.20)	$\mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(D)^3)$	shifted Popov basis

Combining this new approach to compute the $\mathbf{s}$-Popov basis with techniques developed for approximant bases, we also obtain improvements upon previous work when all moduli are powers of X. Compared to [GJV03, Sto06, ZL12], at the cost of one logarithm factor, our algorithm computes the canonical $\mathbf{s}$-Popov basis, supports arbitrary shifts, and covers a more general case since the moduli are not required to be the same power $X^{D/n}$.

Improvement: approximant bases

[GJV03]	$\mathcal{O}(m^{\omega}\mathsf{M}(D/n)\log(D))$	$\mathbf{s}$-minimal, $\mathfrak{m}_i = X^{D/n}$
[Sto06, ZL12]	$\mathcal{O}(m^{\omega}\mathsf{M}(D/m)\log(D))$	$\mathbf{s}$-minimal, $\mathfrak{m}_i = X^{D/n}$, small shifts
Here (Theorem 2.17)	$\mathcal{O}(m^{\omega}\mathsf{M}(D/m)\log(D)^2)$	$\mathbf{s}$-Popov, $\mathfrak{m}_i = X^{D_i}$, $D = \sum_{1\leqslant i\leqslant n} D_i$

⋆⋆⋆

Systems of linear modular equations with arbitrary moduli

Now, we consider the system of Eq. (2) for moduli in $\mathbb{K}[X]$ that are given to us by their coefficients; we do not know their roots, which may not be over $\mathbb{K}$ in general. This makes a difference regarding the available algorithmic techniques: in this context, the bottom-up approach of [VBB92, BL94, BL00] does not work as such. A simple illustration is for a single modular equation: it is unclear how to define subproblems with half the dimension without having access to factors of the modulus.

In the top-down approach, on the other hand, we do not need to know the roots: the equations and moduli give a basis of a module,

$$\mathbf{A} = \begin{bmatrix} \mathbf{I} & [f_{ij}]_{i,j} \\ \mathbf{0} & \operatorname{diag}(\mathfrak{m}_1, \ldots, \mathfrak{m}_n) \end{bmatrix};$$

the sought basis of solutions can be found by computing the shifted Popov form of $\mathbf{A}$ for a well-chosen shift. Yet, the efficiency of the latter computation is not satisfactory with currently known algorithms.

A faster solution is to compute a left kernel for the right part of $\mathbf{A}$ via the algorithm of [ZLS12], based on approximant bases. Still, this becomes less efficient when the shift has large entries or the moduli have non-uniform degrees, and the output basis is not canonical.

One obstacle in both these approaches is that the quotients $(p_1 f_{1i} + \cdots + p_m f_{mi})/\mathfrak{m}_i$ are also computed, for all n equations and all m solutions in the basis. In general, even the number of field elements used to represent these nm quotients may be beyond our target cost bound, especially when n is large with respect to m.

In the next table, we assume $n \in \mathcal{O}(m)$; we write $D_i = \deg(\mathfrak{m}_i)$ and $D = D_1 + \cdots + D_n$.

Improvement: systems of linear modular equations (arbitrary moduli)

Via reduction [GSSV12]	$\tilde{\mathcal{O}}(m^\omega D)$	minimal basis, small shift
Via kernel [ZLS12]	$\tilde{\mathcal{O}}(m^\omega \max_i(D_i))$	minimal basis, small shift
Via kernel [ZLS12]	$\tilde{\mathcal{O}}(m^\omega \lceil D/m \rceil)$	min. basis, shift $\mathbf{0}$, $D_i \approx D/n$
Here (Theorem 2.22)	$\tilde{\mathcal{O}}(m^{\omega-1} D)$	shifted Popov basis
Here (Theorem 2.25)	$\mathcal{O}(m^{\omega-1} \mathsf{M}(D) \log(D))$	a degree-constrained solution

To a small extent, the bottom-up approach can still be used: it suggests a divide-and-conquer approach on the number of equations. At the base case, we thus consider a single equation ($n = 1$) with a modulus of degree D.

Efficient solutions are provided by the linear algebra approach when $D \in \mathcal{O}(m)$, and by approximant basis computations when the *amplitude* $\max(\mathbf{s}) - \min(\mathbf{s})$ of $\mathbf{s} \in \mathbb{Z}^m$ is in $\mathcal{O}(D)$. When this amplitude is large, the shift partially indicates a block-triangular shape of the output basis; this has already been exploited in the context of approximant bases [ZL12, Sec. 6]. We split $\mathbf{s}$ into two parts of half its amplitude, so as to reveal this shape precisely via recursive calls. At the base case of this recursion, shifts have small amplitude and the equation can be solved by approximant bases computation.

To combine the results from recursive calls and obtain the Popov basis for any shift, we manage to exploit our strategy based on finding and using degree information. However, once degrees are known we make use of kernel bases and face the same obstacle as above, leading us to assume $n \in \mathcal{O}(m)$. This assumption also allows us to compute the residual efficiently via polynomial matrix multiplication.

We also provide another algorithm, which relies on the fast structured linear system solver of [BJMS16], with the system being either quasi-Toeplitz or mosaic-Hankel. While this algorithm is faster by some logarithmic factors, we note that it is probabilistic and only returns one solution $(p_1, \ldots, p_m)$ which satisfies prescribed degree constraints.

Concerning the assumption $n \in \mathcal{O}(m)$, as of today it is unclear to us whether and how the cost bound $\tilde{\mathcal{O}}(m^{\omega-1}D)$ can be achieved for an arbitrary number n of equations.

⋆⋆⋆

Multivariate interpolation and list-decoding algorithms

Our results concerning systems of linear modular equations directly apply to problems of multivariate interpolation with multiplicities. Here, we focus on those encountered in list-decoding algorithms; for more general results, we refer to Section 3.1.

The list- and soft-decoding algorithms for Reed-Solomon codes in [GS99, KV03a] rely on finding a bivariate interpolant, as follows. Given some points $\{(x_1, y_1), \ldots, (x_\nu, y_\nu)\}$ in $\mathbb{K}^2$ and integers $\mu_1, \ldots, \mu_\nu$ in $\mathbb{Z}_{>0}$, we look for a nonzero $Q \in \mathbb{K}[X, Y]$ whose degree in Y is less than m and which satisfies

$$Q(x_i, y_i) = 0 \text{ with multiplicity } \mu_i \text{ for } 1 \leqslant i \leqslant \nu, \tag{3}$$

as well as some weighted degree constraint. Here, vanishing with multiplicity means that the shifted polynomial $Q(X + x_i, Y + y_i)$ has no monomial of total degree less than μ_i.

The Y-degree constraint leads to identifying Q with the vector $(p_1, \ldots, p_m) \in \mathbb{K}[X]^m$ such that $Q = \sum_{j<m} p_{j+1}(X)Y^j$. Besides, Eq. (3) can be rewritten as a system of linear modular equations [ZGA11, Zeh13], with moduli that have known roots among $x_1, \ldots, x_\nu$ and known multiplicities related to $\mu_1, \ldots, \mu_\nu$; the sum of their degrees is $D = \mu_1^2 + \cdots + \mu_\nu^2$.

Following the above discussions, a solution Q may be found in $\tilde{\mathcal{O}}(m^{\omega-1}D)$ by computing a shifted minimal basis of the set of $(p_1, \ldots, p_m)$ such that Eq. (3), where the shift is chosen according to the weighted degree constraint and happens to have small entries. In many cases, the dimensions of the system satisfy $n \in \mathcal{O}(m)$ and Q can also be found slightly faster by solving a structured linear system over $\mathbb{K}$; however this may not be the case if there are many repetitions in $x_1, \ldots, x_\nu$, like in soft-decoding.

Improvement: Reed-Solomon list-decoding ($\mu = \mu_1 = \cdots = \mu_\nu$, distinct x_i's)

[Ber11, CH11, CH15]	$\mathcal{O}(m^\omega \mathsf{M}(\mu\nu)\log(\nu))$	deterministic, via row reduction
Here (Corollary 3.1)	$\mathcal{O}(m^{\omega-1}\mathsf{M}(\mu^2\nu)\log(\nu))$	probabilistic, structured system
Here (Corollary 3.2)	$\mathcal{O}(m^{\omega-1}\mathsf{M}(\mu^2\nu)\log(\nu)^2)$	deterministic, via interpolant basis

A similar problem involving more variables arises in the list-decoding of folded Reed-Solomon codes [GR08]. In this context, $Q \in \mathbb{K}[X, Y_1, \ldots, Y_r]$ can be represented by a vector in $\mathbb{K}[X]^m$, where $m = \binom{r+\lambda}{r}$ and λ is known as the *list-size* parameter. We have $\mu_1 = \cdots = \mu_\nu = \mu$, and we obtain a system of linear modular equations with $D = \binom{r+\mu}{r+1}\nu$.

Improvement: folded Reed-Solomon list-decoding

[CH12]	$\mathcal{O}(m^\omega \mathsf{M}(\mu\nu)\log(\nu))$	deterministic, via row reduction
Here (Corollary 3.1)	$\mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(\nu))$	probabilistic, structured system
Here (Corollary 3.2)	$\mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(\nu)^3)$	deterministic, shifted Popov basis

Here, the x_i's are distinct, ensuring that the structured system approach is efficient. However, for a deterministic solution, unlike above, the shift does not have small entries; therefore we rely on our algorithm for computing a Popov basis for an arbitrary shift.

⋆ ⋆ ⋆

Normal forms of polynomial matrices

We consider the computation of the canonical basis of a submodule of $\mathbb{K}[X]^m$ of rank m. This module is specified by any of its bases, forming the rows of an $m \times m$ nonsingular polynomial matrix $\mathbf{A}$; for a given shift $\mathbf{s}$, we want to compute the $\mathbf{s}$-Popov form of $\mathbf{A}$.

Following ideas in [GS11], this problem reduces to that of finding the $\mathbf{s}$-Popov basis of solutions to a system of linear modular equations. In short, the Smith form of $\mathbf{A}$ and a corresponding unimodular multiplier give us the moduli and the equations, respectively. Using algorithms in [Sto03, Gup11] to compute these, and our algorithm for solving the system, we obtain the $\mathbf{s}$-Popov form of $\mathbf{A}$ in $\tilde{\mathcal{O}}(m^\omega d)$ operations, where d is the largest degree of the entries of $\mathbf{A}$. Note that the Smith form algorithm of [Sto03] is probabilistic.

We also design an algorithm which relies on deterministic row reduction [GSSV12] to compute the $\mathbf{s}$-Popov form of $\mathbf{A}$ in $\tilde{\mathcal{O}}(m^\omega d)$ operations if we have a priori knowledge of its column degrees. For shifts such that this canonical form is the Hermite form of $\mathbf{A}$, there is a deterministic algorithm which computes these column degrees efficiently [Zho12]. In this case, we thus have the same cost bound with a deterministic algorithm.

Finally, we show how partial linearization techniques from [GSSV12] can be used to reduce the non-uniformity of the degrees in $\mathbf{A}$. With only a minor increase of m, $\deg(\mathbf{A})$ is decreased to being at most $\lceil D/m \rceil$. Here, D is some bound on $\deg(\det(\mathbf{A}))$ such that D/m is in particular bounded from above by the average row and column degrees of $\mathbf{A}$. Thus, the resulting cost bound $\tilde{\mathcal{O}}(m^\omega \lceil D/m \rceil)$ is in some cases much better than $\tilde{\mathcal{O}}(m^\omega d)$.

Improvement: Shifted Popov and Hermite forms of a matrix ($\star$ = probabilistic)

[GJV03]	$\tilde{\mathcal{O}}(m^\omega d)$	reduced form, small shifts ($\star$)
[GSSV12]	$\tilde{\mathcal{O}}(m^\omega d)$	reduced form, small shifts
[GS11, Gup11]	$\tilde{\mathcal{O}}(m^\omega d)$	Hermite form ($\star$)
Here (Theorem 3.9)	$\tilde{\mathcal{O}}(m^\omega \lceil D/m \rceil)$	Hermite form
Here (Theorem 3.7)	$\tilde{\mathcal{O}}(m^\omega \lceil D/m \rceil)$	shifted Popov form ($\star$)

⋆ ⋆ ⋆

Outline

We now describe the organization of the document, which consists of five parts. The first part introduces our main problems and presents for each of them an overview of our results and a comparison with previous work. Our motivation is that gathering the problems and an overview of contributions in a single part should both simplify the reading and put more emphasis on the strong connections between the problems that are studied. After that, in the next four parts, we give the details of our algorithms: for the multivariate

case, then for systems of linear modular equations with arbitrary moduli, then for the case of known roots, and finally for the computation of normal forms of polynomial matrices.

In the first part, Chapter 1 presents some background on canonical generating sets for $\mathbb{K}[\boldsymbol{X}]$-submodules of $\mathbb{K}[\boldsymbol{X}]$: reduced Gröbner bases in the multivariate case, and shifted Popov bases in the univariate case; we show that in the latter case these notions are equivalent. Furthermore, we emphasize some key ingredients in our algorithms for systems of modular equations. Precisely, we discuss how the shift affects the degree profiles of the computed bases, and we describe our general strategy which consists in recursively finding information on the degrees and then using it to efficiently compute the basis.

In the second chapter, we focus on the main problem in this document: computing reduced Gröbner bases of modules of relations with known multiplication matrices. We consider the general problem as well as the above-mentioned particular cases of approximant bases and of systems of linear modular equations over $\mathbb{K}[X]$, with and without the knowledge of the roots of the moduli. In each situation, we give an overview of our algorithms and we compare our results to previous work.

Finally, in a third chapter, we present consequences of these results. We detail how systems of linear modular equations can be used both to solve problems of multivariate interpolation with multiplicities which arise in decoding algorithms, and to compute shifted Popov forms of polynomial matrices.

Now, we list publications and unpublished reports corresponding to the material in the parts that contain the technical details.

In the second part, Chapter 4 extends the results in [JNSV17, Sec. 7] to the multivariate case, while Chapter 5 is new.

In the third part, the material in Sections 6.2 and 6.3 is from [JNSV16] and [Nei16], respectively, while Section 6.4 presents an algorithm from [BL00]. Chapter 7 is essentially a specialization of the ideas [JNSV16] to the case of approximant bases, taking advantage of the partial linearization techniques from [Sto06] that are available in this context. Chapter 8 is an extended version of [Nei16, Sec. 2]. Chapters 9 and 10 respectively come from [CJN+15] and unpublished notes for the *Lattice and crypto meeting* (ENS de Lyon, March 3, 2016).

In the fourth part, Chapter 11 comes from [CJN+15], while Chapters 12 to 14 gather the results in [JNSV17, JNSV16].

In the last part, Chapter 15 is an extended version of [Nei16, Sec. 3], and the results in Chapter 16 were obtained in collaboration with George Labahn and Wei Zhou [LNZ16].

Part I

Problems and overview of contributions

Contents

Chapter 1 Generating sets of modules over polynomial rings **19**

1.1 Popov bases of modules over univariate polynomial rings 20

1.1.1 Bases and polynomial matrices 20

1.1.2 Row degrees and shifted reduced forms 22

1.1.3 Pivots and shifted Popov forms 27

1.2 Designing fast algorithms for shifted Popov bases 30

1.2.1 Finding and using the minimal degree 30

1.2.2 Size of bases and target costs 35

1.3 Gröbner bases of modules over multivariate polynomial rings 38

1.3.1 Generating sets of ideals and modules 39

1.3.2 Monomial orders and initial module 41

1.3.3 Gröbner bases 46

1.3.4 Link with shifted Popov bases 48

1.3.5 Modules of finite (co)dimension and multiplication matrices . . 50

Chapter 2 Fast computation of relation bases **55**

2.1 Relations or syzygies in finite-dimensional modules 55

2.1.1 Gröbner bases of multivariate modules of relations 55

2.1.2 Univariate case: minimal relation bases 60

2.1.3 Overview of our results 62

2.2 Fast algorithms for dense multiplication matrices 63

2.2.1 Results 63

2.2.2 Overview of our algorithm 65

2.2.3 Change of monomial order for zero-dimensional ideals 68

2.3 Multiplication matrix in nilpotent Jordan form 70

2.3.1 Link with minimal approximant bases 70
2.3.2 Overview of previous work 71
2.3.3 Computing shifted Popov approximant bases 72
2.4 Multiplication matrix in Jordan form 75
2.4.1 Link with minimal interpolant bases 75
2.4.2 Algorithm for almost uniform shifts 76
2.4.3 Computing shifted Popov interpolant bases 78
2.5 Companion-block diagonal multiplication matrix 80
2.5.1 Link with systems of linear modular equations 81
2.5.2 Computing shifted Popov solution bases 82
2.5.3 Computing a solution via structured linear algebra 86

Chapter 3 Impact on related problems **89**
3.1 Multivariate interpolation and list-decoding algorithms 89
3.1.1 Multivariate interpolant with degree constraints 89
3.1.2 List-decoding of (folded) Reed-Solomon codes 92
3.1.3 Computing shifted Popov bases of multivariate interpolants . . 96
3.1.4 Soft-decoding of Reed-Solomon codes 100
3.1.5 General Coppersmith technique over $\mathbb{K}[X]$ 101
3.2 Computing shifted Popov forms of polynomial matrices 105
3.2.1 Overview 105
3.2.2 Computing shifted Popov forms for arbitrary shifts 109
3.2.3 Deterministic computation of Hermite forms and determinants . 110

1

Generating sets of modules over polynomial rings

In this chapter, we present properties about generating sets for modules over polynomial rings, with a view towards algorithms. Informally, a module can be thought of as a vector space where the base ring is not a field; the reader may refer to [DF04, Chapters 10 and 12] for more details. In this document, we will be interested in computing canonical generating sets that have good properties: *Popov bases* in the univariate case and *reduced Gröbner bases* in the multivariate case.

We start with modules over the univariate polynomial ring $\mathbb{K}[X]$, and more precisely, submodules of $\mathbb{K}[X]^m$ for some $m \in \mathbb{Z}_{>0}$. The fact that $\mathbb{K}[X]$ is a principal ideal domain yields good structural properties for these submodules; in particular, they always have a basis, that is, a finite set of generating elements which are $\mathbb{K}[X]$-linearly independent.

In many situations one wants to compute such bases that have some type of minimal degree, for some degree measure to be specified. Working over $\mathbb{K}[X]$, this measure is given through a tuple of integers called *shift*, and the minimality is embodied by the notions of *shifted reduced bases* and *shifted Popov bases*, encompassing the Popov and Hermite forms. We will present these forms and their basic properties.

Then, we present ingredients that are at the core of the efficiency of our algorithms for computing shifted Popov bases. An important aspect in the design and cost analysis of these algorithms is the distribution of the degrees of the elements in these bases, which may be difficult to control in the case of arbitrary shifts. We provide a detailed discussion of some bounds on these degrees.

We also highlight the important role played by the *minimal degree* of shifted Popov bases in our algorithms. We explain how knowing this minimal degree allows us to circumvent the difficulties of controlling the degrees for arbitrary shifts, and we show a general divide-and-conquer approach to find the minimal degree for the kind of problems we will be studying.

After that, we consider the case of a multivariate ring $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X_1, \ldots, X_r]$, with ideals in $\mathbb{K}[\boldsymbol{X}]$ and more generally submodules of $\mathbb{K}[\boldsymbol{X}]^m$. Having $r \geqslant 2$ brings a major difference with the univariate case: most ideals and modules do not have a basis, that is, a set of generating and $\mathbb{K}[\boldsymbol{X}]$-linearly independent elements. Yet, ideals and modules

always have a finite generating set.

Specific such generating sets with useful properties are *Gröbner bases*[1], which depend on the choice of an ordering of the monomials. For a module and some *monomial order*, a canonical form is given by the notion of *reduced Gröbner basis*.

We will introduce Gröbner bases and their basic properties, as well as the notion of *multiplication matrix* which is a central computational tool in this document. We also show that, in the univariate case, any monomial order can be defined by some shift, and reduced Gröbner bases coincide with Popov bases.

1.1 Popov bases of modules over univariate polynomial rings

Here, we focus on the situation where the polynomial ring is univariate. We first recall some nice properties of this case, which naturally lead us to represent bases of modules as univariate polynomial matrices. We also give some examples linked with questions at the core of this thesis. Then, we study degree minimality aspects for bases of these modules, the measure of their elements being given by the *shifted degree*.

In this context, bases that have minimal degree with respect to a given *shift* are said to be in *shifted reduced form*. For a given module and a given shift, among its bases in shifted reduced forms, there is a canonical one which is called the *shifted Popov form*; for a specific shift, this form coincides with the *Hermite form*.

1.1.1 Bases and polynomial matrices

As a preliminary paragraph, we recall some definitions about free modules. For a polynomial ring $\mathcal{R} = \mathbb{K}[X_1, \ldots, X_r]$ with one or several variables, a *free* $\mathcal{R}$-module is a module that has a basis, that is, a generating set which consists of $\mathcal{R}$-linearly independent elements. Then, the *rank* of this $\mathcal{R}$-module is the cardinality of this basis; this is well-defined, since any other basis of this module has the same cardinality [Eis95, Corollary 4.4]. Besides, for such rings $\mathcal{R}$, the rank of an $\mathcal{R}$-module coincides with the maximum number of $\mathcal{R}$-linearly independent elements in this module [DF04, Section 12.1, Proposition 3].

Let us now focus on the univariate case. In what follows, we consider bases of $\mathbb{K}[X]$-submodules of $\mathbb{K}[X]^m$, for some given $m \in \mathbb{Z}_{>0}$. The fact that $\mathbb{K}[X]$ is a principal ideal domain implies many useful properties that we summarize in this section. More details can be found for example in [DF04, Sections 10.3 and 12.1]. A central result is the following [DF04, Section 12.1, Theorem 4].

Lemma 1.1. *Let $\mathcal{M}$ be a $\mathbb{K}[X]$-submodule of $\mathbb{K}[X]^m$. Then,*

(i) $\mathcal{M}$ is free of rank at most m,

(ii) any $\mathbb{K}[X]$-submodule of $\mathcal{M}$ is also free and of rank at most the rank of $\mathcal{M}$.

[1] Gröbner bases are generating sets that are in most cases not linearly independent with respect to the polynomial ring: they are not bases regarding module theory.

Let $\mathcal{M}$ be a $\mathbb{K}[X]$-submodule of $\mathbb{K}[X]^m$. Then $\mathcal{M}$ is free of rank $\rho \leqslant m$, so that any basis of $\mathcal{M}$ can be represented by a $\rho \times m$ matrix over the univariate polynomials, whose rows are the elements of the basis. We denote by $\mathbb{K}[X]^{\rho\times m}$ the set of such matrices, which are called *polynomial matrices*. Hereafter, we see the elements of $\mathcal{M}$ as row vectors in $\mathbb{K}[X]^{1\times m}$, and we identify a basis of $\mathcal{M}$ with the corresponding polynomial matrix. Then, for any basis $\mathbf{A} \in \mathbb{K}[X]^{\rho\times m}$ of $\mathcal{M}$, we have $\mathcal{M} = \{\boldsymbol{\lambda}\mathbf{A}, \boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times\rho}\} = \mathbb{K}[X]^{1\times\rho}\mathbf{A}$. The latter set is also called the *row space* of $\mathbf{A}$.

Example 1.2 (Unimodular matrices). The identity matrix $\mathbf{I}_m \in \mathbb{K}[X]^{m\times m}$ is a basis of the module $\mathbb{K}[X]^m$. Besides, any unit lower triangular matrix

$$\mathbf{T} = \begin{bmatrix} 1 & & & \\ * & 1 & & \\ \vdots & \ddots & \ddots & \\ * & \cdots & * & 1 \end{bmatrix} \in \mathbb{K}[X]^{m\times m}$$

is also a basis of $\mathbb{K}[X]^m$. This can be verified by considering $\mathbf{p} = [p_1, \ldots, p_m] \in \mathbb{K}[X]^{1\times m}$ and solving a $\mathbb{K}[X]$-linear system to determine coefficients $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_m]$ such that $\boldsymbol{\lambda}\mathbf{T} = [p_1, \ldots, p_m]$, noticing that the resulting $\boldsymbol{\lambda}$ has entries in $\mathbb{K}[X]$ due to the special shape of $\mathbf{T}$.

More generally, the bases of $\mathbb{K}[X]^m$ form the set of *unimodular* matrices, defined as follows. A square matrix $\mathbf{U} \in \mathbb{K}[X]^{m\times m}$ is said to be unimodular if $\mathbf{U}$ is nonsingular and its inverse $\mathbf{U}^{-1}$ has entries in $\mathbb{K}[X]$ or, equivalently, if its determinant $\det(\mathbf{U})$ is a nonzero element of the field $\mathbb{K}$. ✍

For any submodule $\mathcal{M} \subseteq \mathbb{K}[X]^m$, two bases of $\mathcal{M}$ are related through a unimodular transformation. Indeed, let $\mathbf{A}$ and $\mathbf{B}$ be two bases of $\mathcal{M}$. Then since $\mathbf{A}$ is a basis, each row of $\mathbf{B}$ is in the row space of $\mathbf{A}$, so there exists a matrix $\mathbf{V} \in \mathbb{K}[X]^{\rho\times\rho}$ such that $\mathbf{B} = \mathbf{V}\mathbf{A}$; similarly, since $\mathbf{B}$ is a basis, there exists $\mathbf{W} \in \mathbb{K}[X]^{\rho\times\rho}$ such that $\mathbf{A} = \mathbf{W}\mathbf{B}$. Thus we have $\mathbf{B} = \mathbf{V}\mathbf{W}\mathbf{B}$, hence $(\mathbf{I}_\rho - \mathbf{V}\mathbf{W})\mathbf{B} = \mathbf{0}$. Since the rows of $\mathbf{B}$ are $\mathbb{K}[X]$-linearly independent by definition, we obtain $\mathbf{V}\mathbf{W} = \mathbf{I}_\rho$. Thus $\mathbf{V}$ and $\mathbf{W}$ are unimodular and inverses of each other. We say that $\mathbf{A}$ and $\mathbf{B}$ are *left-unimodularly equivalent*.

Remark 1.3. In some contexts, or depending on the preferences of the authors, one rather asks that the basis forms the *columns* of an $m \times \rho$ matrix. Then, we have similar notions of *column* space, and *right*-unimodular equivalence. Furthermore, polynomial matrices are sometimes called matrix polynomials, seeing these objects as polynomials with matrix coefficients (while polynomial matrix suggests a matrix with polynomial entries). ☕

We now give examples of $\mathbb{K}[X]$-submodule of $\mathbb{K}[X]^m$. As mentioned above, one is often interested in computing one of its bases that has some kind of minimality: we present the related notion of reduced basis in Section 1.1.2.

Example 1.4 (Constrained bivariate interpolation). Let us consider $\{(x_j, y_j) \in \mathbb{K}^2, 1 \leqslant j \leqslant D\}$ a set of pairwise distinct points in $\mathbb{K}^2$, and fix a Y-degree bound $m \in \mathbb{Z}_{>0}$. Then

$$\{Q \in \mathbb{K}[X, Y] \mid \deg_Y(Q) < m \text{ and } Q(x_j, y_j) = 0 \text{ for all } 1 \leqslant j \leqslant D\}$$

is a free $\mathbb{K}[X]$-module, since it can be seen as a submodule of $\mathbb{K}[X]^m$ by identifying any polynomial Q of Y-degree less than m with its Y-coefficients $Q_0, \ldots, Q_{m-1}$ such that

$Q = \sum_{0 \leqslant j < m} Q_j Y^j$. Besides, it contains the module $M\mathbb{K}[X]^m$ for any nonzero polynomial $M \in \mathbb{K}[X]$ such that $M(x_j) = 0$ for all j; since the rank of $M\mathbb{K}[X]^m$ is m, the rank of the module above is m as well, according to Lemma 1.1. ✍

Example 1.5 (Hermite-Padé approximation). Let $\mathbf{F} \in \mathbb{K}[X]^{m \times 1}$ be a polynomial vector, and let D be a positive integer. Then

$$\{\mathbf{p} \in \mathbb{K}[X]^{1 \times m} \mid \mathbf{pF} = \mathbf{0} \bmod X^D\}$$

is a free $\mathbb{K}[X]$-module of rank m. Indeed, this set is included in $\mathbb{K}[X]^{1 \times m}$ and contains $X^D \mathbb{K}[X]^{1 \times m}$, which are both of rank m. ✍

Example 1.6 (Row space of a matrix). Consider a polynomial matrix $\mathbf{A} \in \mathbb{K}[X]^{k \times m}$. Then the row space of $\mathbf{A}$, which is the set $\mathbb{K}[X]^{1 \times k}\mathbf{A}$, is a free $\mathbb{K}[X]$-module of rank $\rho \leqslant \min(k, m)$. This number ρ is also called the rank of the matrix $\mathbf{A}$. If $\rho = \min(k, m)$, then $\mathbf{A}$ is said to have *full rank*; if furthermore $k = m$ (that is, $\mathbf{A}$ is a square matrix), then $\mathbf{A}$ is said to be nonsingular. Notice that $\mathbf{A}$ is nonsingular if and only if $\det(\mathbf{A})$ is a nonzero polynomial. ✍

Example 1.7 (Kernel of a matrix). Let $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ be a polynomial matrix. Then the *kernel* of $\mathbf{F}$ is the set

$$\{\mathbf{p} \in \mathbb{K}[X]^{1 \times m} \mid \mathbf{pF} = \mathbf{0}\}.$$

It is a free $\mathbb{K}[X]$-module of rank $m - \rho$, where ρ is the rank of $\mathbf{F}$. ✍

In the case of modules of rank m, the next result is often helpful to prove that some given elements generate the module.

Lemma 1.8. *Let $\mathcal{M}$ be a $\mathbb{K}[X]$-submodule of $\mathbb{K}[X]^m$ of rank m and let $\mathbf{A} \in \mathbb{K}[X]^{m \times m}$ be a basis of $\mathcal{M}$. Then, for any nonsingular matrix $\mathbf{B} \in \mathbb{K}[X]^{m \times m}$ whose rows are in $\mathcal{M}$, we have $\deg(\det(\mathbf{B})) \geqslant \deg(\det(\mathbf{A}))$; if furthermore $\deg(\det(\mathbf{B})) = \deg(\det(\mathbf{A}))$ then $\mathbf{B}$ is also a basis of $\mathcal{M}$.*

Proof. Since the rows of $\mathbf{B}$ are in $\mathcal{M}$, there is $\mathbf{U} \in \mathbb{K}[X]^{m \times m}$ such that $\mathbf{B} = \mathbf{UA}$. Having $\mathbf{A}$ and $\mathbf{B}$ nonsingular implies that $\mathbf{U}$ is nonsingular too, and taking the determinant yields $\deg(\det(\mathbf{B})) = \deg(\det(\mathbf{U})) + \deg(\det(\mathbf{A})) \geqslant \deg(\det(\mathbf{A}))$. Now, in the case of equality, this implies that $\deg(\det(\mathbf{U})) = 0$, and therefore $\mathbf{U}$ is unimodular. ☛

1.1.2 Row degrees and shifted reduced forms

One is often interested in computing bases whose elements have small degrees. For example, when the module is described by a set of equations as in the examples of bivariate interpolation and Hermite-Padé approximation above, it is usually asked to compute one element of the module which satisfies some degree constraints. With currently known algorithms, the fastest way to obtain such an element is to compute a whole basis of the module which has some type of minimal degrees, and contains in particular a small degree element. Such a basis is said to be in *reduced* form; we give precise definitions below.

There is also another important reason for requiring that the computed bases have small degrees: as a matter of efficiency of the algorithms, setting more constraints on the output also allows us to better control the size of the objects that we manipulate during the computation. We will come back to this point in Sections 1.1.3 and 1.2.2.

Shifted degrees and degree constraints. We now explain how we measure the degree of elements of $\mathbb{K}[X]^{1\times m}$, which we call (polynomial) row vectors. In the simplest case, we use a straightforward extension of the degree of univariate polynomials: the *degree* of $\mathbf{p} = [p_j]_j \in \mathbb{K}[X]^{1\times m}$ is $\operatorname{rdeg}(\mathbf{p}) = \max_{1\leqslant j\leqslant m} \deg(p_j)$. Here, we follow the convention that the degree of a zero polynomial is $\deg(0) = -\infty$, and we extend it to row vectors. The degree of a column vector $\mathbf{p} \in \mathbb{K}[X]^{m\times 1}$ is defined similarly and denoted by $\operatorname{cdeg}(\mathbf{p})$.

More generally, we will measure the degree by considering *shifted degrees*. Hereafter, a *shift* is a tuple of integers $\mathbf{s} = (s_1, \ldots, s_m) \in \mathbb{Z}^m$ which specifies degree weights on the entries of row vectors. Then, the $\mathbf{s}$-*degree* of $\mathbf{p} = [p_j]_j \in \mathbb{K}[X]^{1\times m}$ is

$$\operatorname{rdeg}_{\mathbf{s}}(\mathbf{p}) = \max_{1\leqslant j\leqslant m} (\deg(p_j) + s_j).$$

For example, the degree of $\mathbf{p}$ as defined in the previous paragraph is its $\mathbf{0}$-degree for the *uniform* shift $\mathbf{0} = (0, \ldots, 0)$.

Remark 1.9. We allow $\mathbf{s}$ to have negative entries, and therefore the $\mathbf{s}$-degree of a nonzero row vector may be negative. One reason behind this choice lies in the following remark. In many situations, one wants to find some element $\mathbf{p} = [p_j]_j$ in a given module with prescribed degree constraints. Namely, some positive integers $(N_j)_j$ are given, and one requires that $\deg(p_j) < N_j$ for all j. Then, we can directly relate this to shifted row degrees by noticing that this is equivalent to requiring that $\operatorname{rdeg}_{-\mathbf{N}}(\mathbf{p}) < 0$, where $\mathbf{N} = (N_j)_j$. ☕

Up to a change of sign, this notion of $\mathbf{s}$-degree is equivalent to the one introduced in [BLV06] for matrix normal forms, and also to the notion of *defect* from [BL94, Definition 3.1] commonly used in the rational approximation literature. The usefulness of shifted degrees can be seen from their applications. First, they allow us to take degree constraints into account in many problems studied in this thesis, such as Hermite-Padé and M-Padé approximations [VBB92, Bec92, BL94] (see also Sections 2.3 and 2.4.1), and the interpolation step of the list-decoding and soft-decoding algorithms for Reed-Solomon codes [GS99, KV03a] (see also Sections 3.1.2 and 3.1.4). Second, they also play a central role in the design of fast algorithms for polynomial matrices, such as for computing minimal kernel bases [ZLS12] or for matrix inversion [ZLS15].

This notion of shifted degree can then be naturally extended to polynomial matrices: for $\mathbf{A} \in \mathbb{K}[X]^{k\times m}$, its *row degree* is the tuple $\operatorname{rdeg}(\mathbf{A}) = (d_1, \ldots, d_k) \in \mathbb{Z}^k_{\geqslant 0}$, where for all i, d_i is the row degree of the row i of $\mathbf{A}$. In what follows, we use the notations $\mathbf{A}_{i,*}$ and $\mathbf{A}_{*,j}$ to refer to the row i and column j of a matrix $\mathbf{A}$, respectively. Then, the $\mathbf{s}$-*row degree* of $\mathbf{A}$ is $\operatorname{rdeg}_{\mathbf{s}}(\mathbf{A}) = (d_1, \ldots, d_k)$, with $d_i = \operatorname{rdeg}_{\mathbf{s}}(\mathbf{A}_{i,*})$ for all i; and the notion of *shifted column degree* $\operatorname{cdeg}_{\mathbf{t}}(\mathbf{A})$ for $\mathbf{t} \in \mathbb{Z}^k$ is defined similarly. Besides, we write $\deg(\mathbf{A})$ for the *degree* of $\mathbf{A}$, which is the largest degree of its entries: $\deg(\mathbf{A}) = \max(\operatorname{rdeg}(\mathbf{A})) = \max(\operatorname{cdeg}(\mathbf{A}))$.

When dealing with shifts, one can often think in terms of the case of the uniform shift by means of the *shift matrix* $\mathbf{X}^{\mathbf{s}}$. For a shift $\mathbf{s} = (s_1, \ldots s_m) \in \mathbb{Z}^m$, we denote by $\mathbf{X}^{\mathbf{s}}$ the $m \times m$ diagonal matrix with diagonal entries $X^{s_1}, \ldots, X^{s_m}$. Then, if $\mathbf{s}$ has nonnegative entries, the $\mathbf{s}$-row degree of $\mathbf{A}$ is $\operatorname{rdeg}_{\mathbf{s}}(\mathbf{A}) = \operatorname{rdeg}(\mathbf{A}\mathbf{X}^{\mathbf{s}})$. In fact, in some cases it will be convenient to allow negative powers of X to simplify the presentation. For example, we define the $\mathbf{s}$-*leading matrix* of $\mathbf{A} \in \mathbb{K}[X]^{k\times m}$ as the matrix $\operatorname{lm}_{\mathbf{s}}(\mathbf{A}) \in \mathbb{K}^{k\times m}$ whose entries

are the coefficients of degree zero of $\mathbf{X}^{-\mathrm{rdeg}_{\mathbf{s}}(\mathbf{A})}\mathbf{A}\mathbf{X}^{\mathbf{s}}$. Notice that the latter matrix has only coefficients of nonpositive degree.

Shifted reduced forms. Now, for a $\mathbb{K}[X]$-submodule $\mathcal{M}$ of $\mathbb{K}[X]^m$, we are interested in bases of $\mathcal{M}$ that have a specific form: as already stated, our main motivation is to consider bases that have some type of minimal shifted row degree. We give the following definition, which focuses on this minimality, and we then state some properties that are equivalent to this definition. For early presentations of the notion of reduced form, the reader may refer to [Wol74, Section 2.5], [Kai80, Section 6.3]; more details about *shifted* reduced forms can be found in [VBB92, BLV99] and [Zho12, Chapter 2].

Definition 1.10 (Reduced form)**.** *Let $\rho \leqslant m$ and let $\mathbf{R} \in \mathbb{K}[X]^{\rho\times m}$ have full rank. Then $\mathbf{R}$ is said to be* row reduced *if $\mathrm{rdeg}(\mathbf{R}) \leqslant \mathrm{rdeg}(\mathbf{U}\mathbf{R})$ for all unimodular matrices $\mathbf{U} \in \mathbb{K}[X]^{\rho\times\rho}$. More generally, for a shift $\mathbf{s} \in \mathbb{Z}^m$, $\mathbf{R}$ is said to be* $\mathbf{s}$-row reduced *if $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{R}) \leqslant \mathrm{rdeg}_{\mathbf{s}}(\mathbf{U}\mathbf{R})$ for all unimodular matrices $\mathbf{U} \in \mathbb{K}[X]^{\rho\times\rho}$. In both these inequalities, the tuples are first sorted in non-decreasing order and then compared lexicographically.*

In particular, this notion is invariant under permutation of the rows of the matrix, and all bases of a module that are in $\mathbf{s}$-reduced form must have the same $\mathbf{s}$-row degree up to permutation. The notion of column reducedness is defined similarly; since in what follows we will mostly encounter row reduced forms, unless the context makes it ambiguous we will simply write reduced for *row* reduced.

Besides, adding a constant $c \in \mathbb{Z}$ to each entry of $\mathbf{s}$ does not change the property of $\mathbf{s}$-reducedness. Indeed, writing $\mathbf{s}+c$ for this operation of adding a constant to a tuple, we have $\mathrm{rdeg}_{\mathbf{s}+c}(\mathbf{A}) = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{A}) + c$ for any matrix $\mathbf{A}$. In particular, one may always think in terms of a shift with nonnegative entries by considering $\mathbf{s} - \min(\mathbf{s})$ instead of $\mathbf{s}$. For a shift $\mathbf{s}$ with nonnegative entries, $\mathbf{R}$ is $\mathbf{s}$-reduced if and only if $\mathbf{R}\mathbf{X}^{\mathbf{s}}$ is reduced.

Definition 1.10 essentially tells us that the elements of a basis in shifted reduced form have minimal degrees, in terms of the row degree defined above. There are several equivalent definitions, which we list now, and which will be helpful for designing algorithms and writing proofs. In what follows, for any tuple of integers $\mathbf{s} \in \mathbb{Z}^m$, we denote by $|\mathbf{s}|$ the sum of the entries of $\mathbf{s}$.

Theorem 1.11. *Let $\rho \leqslant m$, let $\mathbf{R} \in \mathbb{K}[X]^{\rho\times m}$ have full rank, and let $\mathbf{s} \in \mathbb{Z}^m$. The following assertions are equivalent.*

(i) $\mathbf{R}$ *is* $\mathbf{s}$*-reduced.*

(ii) For all row vectors $\boldsymbol{\lambda} = \begin{bmatrix}\boldsymbol{\lambda_1} & 1 & \boldsymbol{\lambda_2}\end{bmatrix} \in \mathbb{K}[X]^{1\times\rho}$ with the entry 1 at index i, we have $\mathrm{rdeg}_{\mathbf{s}}(\boldsymbol{\lambda}\mathbf{R}) \geqslant \mathrm{rdeg}_{\mathbf{s}}(\mathbf{R}_{i,*})$.

(iii) The $\mathbf{s}$*-leading matrix* $\mathrm{lm}_{\mathbf{s}}(\mathbf{R})$ *has full rank.*

(iv) Predictable degree property: *for all $\boldsymbol{\lambda} = [\lambda_i]_i \in \mathbb{K}[X]^{1\times\rho}$, we have*

$$\mathrm{rdeg}_{\mathbf{s}}(\boldsymbol{\lambda}\mathbf{R}) = \max\{\deg(\lambda_i) + \mathrm{rdeg}_{\mathbf{s}}(\mathbf{R}_{i,*}),\ 1 \leqslant i \leqslant \rho\} = \mathrm{rdeg}_{\mathbf{d}}(\boldsymbol{\lambda}),$$

where $\mathbf{d} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{R})$.

(v) (Only in the square case, $\rho = m$.) We have $|\mathrm{rdeg}_{\mathbf{s}}(\mathbf{R})| = \deg(\det(\mathbf{R})) + |\mathbf{s}|$.

Proof. In this proof, we write $d_i = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{R}_{i,*})$ for all i, that is, $\mathbf{d} = (d_1, \ldots, d_\rho)$. Then $\mathrm{lm}_{\mathbf{s}}(\mathbf{R})$ is formed by the coefficients of degree 0 of the matrix $\mathbf{X}^{-\mathbf{d}}\mathbf{R}\mathbf{X}^{\mathbf{s}}$, whose coefficients are all of nonpositive degree.

$\neg(ii) \Rightarrow \neg(i)$. Suppose that there is a row $= \begin{bmatrix}\boldsymbol{\lambda_1} & 1 & \boldsymbol{\lambda_2}\end{bmatrix} \in \mathbb{K}[X]^{1\times\rho}$ with 1 at index i and such that $\mathrm{rdeg}_{\mathbf{s}}(\boldsymbol{\lambda}\mathbf{R}) < d_i$. Considering the unimodular transformation

$$\mathbf{U} = \begin{bmatrix}\mathbf{I}_{i-1} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\lambda_1} & 1 & \boldsymbol{\lambda_2} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_{\rho-i}\end{bmatrix} \in \mathbb{K}[X]^{\rho\times\rho}, \tag{1.1}$$

we have that $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{U}\mathbf{R})$ and $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{R})$ are identical except for their ith entries, which are $\mathrm{rdeg}_{\mathbf{s}}(\boldsymbol{\lambda}\mathbf{R})$ and d_i, respectively. Thus, our assumption that $\mathrm{rdeg}_{\mathbf{s}}(\boldsymbol{\lambda}\mathbf{R}) < d_i$ implies that $\mathbf{R}$ is not $\mathbf{s}$-reduced.

$\neg(iii) \Rightarrow \neg(ii)$. Suppose that $\mathrm{lm}_{\mathbf{s}}(\mathbf{R})$ does not have full rank. Therefore, there exists a nontrivial linear combination of the rows of $\mathrm{lm}_{\mathbf{s}}(\mathbf{R})$ which is zero, that is, $\boldsymbol{\mu}\mathrm{lm}_{\mathbf{s}}(\mathbf{R}) = 0$ for some nonzero $\boldsymbol{\mu} = [\mu_i]_i \in \mathbb{K}^{1\times\rho}$. Let i be an index such that d_i is maximal with $\mu_i \neq 0$. Then, we consider $\boldsymbol{\lambda} = \mu_i^{-1}\boldsymbol{\mu}\mathbf{X}^{d_i-\mathbf{d}} \in \mathbb{K}[X]^{1\times\rho}$: by choice of i, this row vector is over the polynomials (we recall that $d_i - \mathbf{d}$ stands for the tuple $-\mathbf{d}$ with the constant d_i added to every entry). We remark that $\boldsymbol{\lambda}$ has coefficient 1 at index i, and furthermore, we have $\mathrm{rdeg}_{\mathbf{s}}(\boldsymbol{\lambda}\mathbf{R}) = \mathrm{rdeg}(\boldsymbol{\lambda}\mathbf{R}\mathbf{X}^{\mathbf{s}}) < d_i = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{R}_{i,*})$. Indeed, $\boldsymbol{\lambda}\mathbf{R}\mathbf{X}^{\mathbf{s}} = \mu_i^{-1}\mathbf{X}^{d_i}\boldsymbol{\mu}\mathbf{X}^{-\mathbf{d}}\mathbf{R}\mathbf{X}^{\mathbf{s}}$, and the polynomial row vector $\boldsymbol{\mu}\mathbf{X}^{-\mathbf{d}}\mathbf{R}\mathbf{X}^{\mathbf{s}}$ has only coefficients of (strictly) negative degree since its coefficient of degree 0 is $\boldsymbol{\mu}\mathrm{lm}_{\mathbf{s}}(\mathbf{R}) = 0$.

$(iii) \Rightarrow (iv)$. Assume that $\mathrm{lm}_{\mathbf{s}}(\mathbf{R})$ has full rank. We always have (iv) for $\boldsymbol{\lambda} = 0$, so we consider the case $\boldsymbol{\lambda} \neq 0$. Let $t = \mathrm{rdeg}_{\mathbf{d}}(\boldsymbol{\lambda})$. We have $\mathrm{rdeg}_{\mathbf{s}}(\boldsymbol{\lambda}\mathbf{R}) \leqslant t$, and our goal is to prove that actually the equality holds. This follows from the fact that the coefficient of degree 0 of the row vector $X^{-t}\boldsymbol{\lambda}\mathbf{R}\mathbf{X}^{\mathbf{s}} = X^{-t}\boldsymbol{\lambda}\mathbf{X}^{\mathbf{d}}\mathbf{X}^{-\mathbf{d}}\mathbf{R}\mathbf{X}^{\mathbf{s}}$ is $\mathrm{lm}_{\mathbf{d}}(\boldsymbol{\lambda})\mathrm{lm}_{\mathbf{s}}(\mathbf{R})$, which is nonzero since $\boldsymbol{\lambda}$ is nonzero and $\mathrm{lm}_{\mathbf{s}}(\mathbf{R})$ has full row rank.

$\neg(i) \Rightarrow \neg(iv)$. Suppose that (i) does not hold: there exists $\mathbf{U} \in \mathbb{K}[X]^{\rho\times\rho}$ unimodular such that $\mathbf{d}' = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{U}\mathbf{R})$ does not satisfy $\mathbf{d} \leqslant \mathbf{d}'$ (where the tuples are first sorted and then compared lexicographically). Up to row permutations, we can assume without loss of generality that $\mathbf{d} = (d_1, \ldots, d_\rho)$ and $\mathbf{d}' = (d'_1, \ldots, d'_\rho)$ are in non-decreasing order. Then there exists $i \in \{1, \ldots, \rho\}$ such that $d'_i < d_i$ and $d'_j = d_j$ for all $j < i$. Therefore, writing

$$\mathbf{U} = \begin{bmatrix}\mathbf{U}_{11} & \mathbf{U}_{12} \\ \mathbf{U}_{21} & \mathbf{U}_{22}\end{bmatrix} \quad \text{with } \mathbf{U}_{11} \in \mathbb{K}[X]^{i\times(i-1)} \text{ and } \mathbf{U}_{22} \in \mathbb{K}[X]^{(\rho-i)\times(\rho-i+1)},$$

all rows in $\begin{bmatrix}\mathbf{U}_{11} & \mathbf{U}_{12}\end{bmatrix}\mathbf{R}$ have $\mathbf{s}$-degree less than d_i, and thus less than all of $d_i, \ldots, d_\rho$. On the other hand, since $\mathbf{U}$ is nonsingular, we have that $\mathbf{U}_{12}$ is nonzero. Let us denote by (j, k) the index of a nonzero entry of $\mathbf{U}$ located in $\mathbf{U}_{12}$. Then, considering the row vector $\boldsymbol{\lambda} = \mathbf{U}_{j,*} = [\lambda_t]_{1\leqslant t\leqslant\rho}$, we obtain that $d'_j = \mathrm{rdeg}_{\mathbf{s}}(\boldsymbol{\lambda}\mathbf{R}) < d_k \leqslant \deg(\lambda_k) + \mathrm{rdeg}_{\mathbf{s}}(\mathbf{R}_{k,*})$.

$(iii) \Leftrightarrow (v)$. By definition of the determinant, $\deg(\det(\mathbf{R})) + |\mathbf{s}| = \deg(\det(\mathbf{R}\mathbf{X}^{\mathbf{s}})) \leqslant |\mathrm{rdeg}_{\mathbf{0}}(\mathbf{R}\mathbf{X}^{\mathbf{s}})| = |\mathrm{rdeg}_{\mathbf{s}}(\mathbf{R})|$. This inequality may be strict if some cancellations of high-degree terms in $\mathbf{R}$ occur: this is precisely what is forbidden by the fact that $\mathrm{lm}_{\mathbf{s}}(\mathbf{R}) = \mathrm{lm}_{\mathbf{0}}(\mathbf{R}\mathbf{X}^{\mathbf{s}})$ has full rank. In fact, the coefficient of $\det(\mathbf{R}\mathbf{X}^{\mathbf{s}})$ of degree $|\mathrm{rdeg}_{\mathbf{s}}(\mathbf{R})|$ is precisely $\det(\mathrm{lm}_{\mathbf{s}}(\mathbf{R}))$, hence the equivalence. ∎

The item (*iii*) in the above theorem is often chosen as a definition of row reducedness, as in [VBB92, BLV99, GJV03]; the item (*v*) is used as a definition in [Kai80, Section 6.3.2]. The predictable degree property in item (*iv*), which was stated in [For75] (see also [Kai80, Theorem 6.3-13]), is quite useful when one has to study the degrees in expressions which involve shifted reduced matrices. A direct consequence is the following. Let $\mathcal{M}$ be a submodule of $\mathbb{K}[X]^m$, and let $\mathbf{R}$ be a basis of $\mathcal{M}$ in $\mathbf{s}$-reduced form for some $\mathbf{s} \in \mathbb{Z}^m$. Then, any $\mathbf{p} \in \mathcal{M}$ has $\mathbf{s}$-degree at least $\min(\mathrm{rdeg}_{\mathbf{s}}(\mathbf{R}))$.

In other words, an element of $\mathcal{M}$ of minimal $\mathbf{s}$-degree can be found as a row of $\mathbf{R}$ of minimal $\mathbf{s}$-degree. In particular, when a problem asks to compute $\mathbf{p} = [p_j]_j \in \mathcal{M}$ with the degree constraints $\deg(p_j) < N_j$ for all j, such a $\mathbf{p}$ can be found among the rows of a basis of $\mathcal{M}$ in $-\mathbf{N}$-reduced form, where $\mathbf{N} = (N_j)_j$ (unless the problem has no solution).

Example 1.12. Let us consider the finite field $\mathbb{K} = \mathbb{Z}/7\mathbb{Z}$ with 7 elements, and choose $D = 4$ points $\{(x_i, y_i),\ 1 \leqslant i \leqslant 4\}$ in $\mathbb{K}^2$ with $(x_i)_i = (2, 1, 4, 6)$ and $(y_i)_i = (6, 2, 1, 6)$.

We will give shifted reduced bases for the $\mathbb{K}[X]$-module $\mathcal{M}$ of bivariate polynomials $Q \in \mathbb{K}[X, Y]$ such that $\deg_Y(Q) < m = 2$ and $Q(x_i, y_i) = 0$ for $1 \leqslant i \leqslant 4$. Then, identifying a polynomial of the form $Q_0(X)+Q_1(X)Y$ with the polynomial vector $[Q_0(X)\ \ Q_1(X)]$, we see $\mathcal{M}$ as a $\mathbb{K}[X]$-submodule of $\mathbb{K}[X]^2$; its rank is $m = 2$.

Defining the polynomials $M = (X-2)(X-1)(X-4)(X-6) = X^4 + X^3 + 6X + 6$ and the interpolant $L = 5X^3 + 6X^2 + 5$ such that $L(x_i) = y_i$ for $1 \leqslant i \leqslant 4$, the matrix

$$\mathbf{P} = \begin{bmatrix} M & 0 \\ -L & 1 \end{bmatrix} = \begin{bmatrix} X^4 + X^3 + 6X + 6 & 0 \\ 2X^3 + X^2 + 2 & 1 \end{bmatrix}$$

is a basis of $\mathcal{M}$. This follows from the remark that a bivariate polynomial Q is such that $Q(x_i, y_i) = 0$ for $1 \leqslant i \leqslant 4$ if and only if Q belongs to the ideal $\langle M, Y - L\rangle$.

The shifted leading matrices of $\mathbf{P}$ for the shift $\mathbf{s} = (0, 3)$ and the uniform shift are

$$\mathrm{lm}_{(0,3)}(\mathbf{P}) = \begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix} \quad \text{and} \quad \mathrm{lm}_{\mathbf{0}}(\mathbf{P}) = \begin{bmatrix} 1 & 0 \\ 2 & 0 \end{bmatrix};$$

therefore $\mathbf{P}$ is a $(0, 3)$-reduced basis of $\mathcal{M}$, but is not $\mathbf{0}$-reduced.

Furthermore, the matrix

$$\mathbf{A} = \begin{bmatrix} 1 & 3X + 5 \\ X & 3X^2 + 5X + 1 \end{bmatrix} \mathbf{P} = \begin{bmatrix} 5X^2 + 5X + 2 & 3X + 5 \\ 6X^2 + 2X + 2 & 3X^2 + 5X + 1 \end{bmatrix}$$

is left-unimodularly equivalent to $\mathbf{P}$ and has a full rank $\mathbf{0}$-leading matrix: $\mathbf{A}$ is a $\mathbf{0}$-reduced basis of $\mathcal{M}$. ✍

Existence and computation. Most importantly, for a given $\mathbb{K}[X]$-submodule $\mathcal{M}$ of $\mathbb{K}[X]^m$ and a given shift $\mathbf{s} \in \mathbb{Z}^m$, it is known that $\mathcal{M}$ admits a basis in $\mathbf{s}$-reduced form. In fact, if $\mathbf{A} \in \mathbb{K}[X]^{k\times m}$ is a generating set for $\mathcal{M}$ for some integer k, then there are algorithms to compute from $\mathbf{A}$ an $\mathbf{s}$-reduced basis $\mathbf{R}$ of $\mathcal{M}$.

Such an algorithm is presented in [Wol74, Theorem 2.5.7]; it iteratively performs elementary operations on $\mathbf{A}$ with unimodular transformations of the form of $\mathbf{U}$ in Eq. (1.1), making at each step some progress towards obtaining a $\mathbf{s}$-reduced form. Although in the

latter reference the algorithm is presented for the uniform shift $\mathbf{s} = \mathbf{0}$, adapting it to work with an arbitrary shift $\mathbf{s}$ is straightforward. This reduction is done in time polynomial in m, k, and $\deg(\mathbf{A})$.

Fast algorithms for the computation of a shifted reduced basis of $\mathcal{M}$ from a known generating set are presented with more details in Section 3.2, where we also discuss the problem of computing a canonical reduced form called the *shifted Popov form*.

1.1.3 Pivots and shifted Popov forms

A particular kind of shifted reduced form is the *shifted Popov form*, which includes the *Popov form* [Pop72, Kai80] and the *Hermite form* [Her51] for specific shifts. In this section, we define these forms and give their basic properties. There are two main reasons that motivate us to study them. First, the shifted Popov form is a *normal form*: it is uniquely defined in terms of the module and the shift. Second, shifted Popov forms have a degree structure which allows us to compute them more efficiently than shifted reduced forms in general; this point will be discussed more in depth in Section 1.2.

A basis is in $\mathbf{s}$-Popov form when it simultaneously satisfies two degree minimality requirements: roughly, it should both be $\mathbf{s}$-row reduced and have columns of minimal $\mathbf{0}$-degree. We should stress that these two requirements are not of the same nature, since both involve unimodular transformations on the rows only.

Pivots. We first present the notion of pivot [Kai80, Section 6.7.2], and specific shifted reduced forms that are more restrictive than shifted reduced forms but are not yet the canonical shifted Popov forms.

Definition 1.13 (Pivot)**.** *Let* $\mathbf{p} = [p_j]_j \in \mathbb{K}[X]^{1\times m}$ *be nonzero and let* $\mathbf{s} \in \mathbb{Z}^m$*. The* $\mathbf{s}$-pivot index *of* $\mathbf{p}$ *is the largest index* j *such that* $\operatorname{rdeg}_{\mathbf{s}}(\mathbf{p}) = \deg(p_j) + s_j$*. Then we call* p_j *and* $\deg(p_j)$ *the* $\mathbf{s}$-pivot entry *and the* $\mathbf{s}$-pivot degree *of* $\mathbf{p}$*.*

Adding a constant to the entries of $\mathbf{s}$ does not change the notion of $\mathbf{s}$-pivot. Thus, like for $\mathbf{s}$-reducedness, one may assume $\min(\mathbf{s}) = 0$ without loss of generality. In connection with the notion of pivot, one can define the weak Popov form [MS03b] and the quasi Popov form [BLV99], which we call here ordered weak Popov form.

Definition 1.14 ((Ordered) weak Popov form)**.** *Let* $\rho \leqslant m$*, let* $\mathbf{P} \in \mathbb{K}[X]^{\rho\times m}$ *have full rank, and let* $\mathbf{s} \in \mathbb{Z}^m$*. Then,* $\mathbf{P}$ *is in* $\mathbf{s}$-weak Popov form *if the* $\mathbf{s}$*-pivot indices of its rows are pairwise distinct. Furthermore,* $\mathbf{P}$ *is in* $\mathbf{s}$-ordered weak Popov form *if the sequence of the* $\mathbf{s}$*-pivot indices of its rows is strictly increasing.*

In other words, $\mathbf{P}$ is in $\mathbf{s}$-ordered weak Popov form if and only if the $\mathbf{s}$-leading matrix of $\mathbf{P}$ is in some type of (non-reduced) row echelon form[2]. In particular, $\operatorname{lm}_{\mathbf{s}}(\mathbf{P})$ contains an $\rho \times \rho$ invertible lower triangular submatrix, and $\mathbf{P}$ is therefore $\mathbf{s}$-reduced. Moreover, any matrix in $\mathbf{s}$-weak Popov form can be transformed into an $\mathbf{s}$-ordered weak Popov form by a permutation of its rows; thus a matrix in $\mathbf{s}$-weak Popov form is in particular $\mathbf{s}$-reduced.

[2]Precisely, it is the transpose of a matrix in *column echelon form* as in [BLV06, Definition 2.2].

Remark 1.15. Here, quasi Popov forms [BLV99] will rather be called *ordered weak* Popov forms to avoid that the reader possibly get the feeling that the difference between them and Popov forms is minor: this is not the case, computationally speaking. Indeed, going from a weak Popov form to an ordered one is a straightforward task: one just has to locate the pivots and to permute the rows of the matrix accordingly; however, going from an ordered weak Popov form to its Popov form is much more involved (see [SS11]).

Now, we extend the notion of pivots to matrices; while it could be defined in general, we will only use it for matrices in shifted weak Popov form.

Definition 1.16 (Pivot of a matrix)**.** *Let $\rho \leqslant m$, let $\mathbf{s} \in \mathbb{Z}^m$, and let $\mathbf{P} \in \mathbb{K}[X]^{\rho \times m}$ be in $\mathbf{s}$-weak Popov form. Then, the* $\mathbf{s}$-pivot index *and the* $\mathbf{s}$-pivot degree *of $\mathbf{P}$ are the tuples of integers $(j_1, \ldots, j_\rho)$ and $(\delta_1, \ldots, \delta_\rho)$ where j_i is the $\mathbf{s}$-pivot index of $\mathbf{P}_{i,*}$ and δ_i is the $\mathbf{s}$-pivot degree of $\mathbf{P}_{i,*}$ for $1 \leqslant i \leqslant \rho$.*

Considering the shift $\hat{\mathbf{s}} = (s_{j_1}, \ldots, s_{j_\rho}) \in \mathbb{Z}^\rho$, which is a permuted subshift of $\mathbf{s}$, we have $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}) = \hat{\mathbf{s}} + \boldsymbol{\delta}$. The notions of pivot indices and pivot degrees for weak Popov matrices refine that of row degrees for reduced matrices, by providing us with a stronger degree minimality property than that used as a definition of shifted reduced forms. This property, stated in the next lemma and illustrated in Example 1.18, has consequences that are presented in Section 1.2 and are exploited to obtain efficiency in several algorithms in this document.

Lemma 1.17. *Let $\rho \leqslant m$, let $\mathbf{s} \in \mathbb{Z}^m$, let $\mathbf{P} \in \mathbb{K}[X]^{\rho \times m}$ be in $\mathbf{s}$-weak Popov form, and let $(j_1, \ldots, j_\rho)$ and $(\delta_1, \ldots, \delta_\rho)$ be the $\mathbf{s}$-pivot index and degree of $\mathbf{P}$. Let $\mathbf{p} \in \mathbb{K}[X]^{1 \times m}$ be a nonzero vector in the row space of $\mathbf{P}$. Then, the $\mathbf{s}$-pivot index of $\mathbf{p}$ is j_i for some $1 \leqslant i \leqslant \rho$, and $\mathbf{p}$ has $\mathbf{s}$-pivot degree at least δ_i.*

Proof. Up to permuting the rows of $\mathbf{P}$, we assume without loss of generality that $\mathbf{P}$ is in $\mathbf{s}$-ordered weak Popov form, that is, $j_1 < \cdots < j_\rho$. Let $\mathbf{d} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}) = (s_{j_1} + \delta_1, \ldots, s_{j_\rho} + \delta_\rho)$, and let $\boldsymbol{\lambda} \in \mathbb{K}[X]^{1 \times m}$ be such that $\mathbf{p} = \boldsymbol{\lambda}\mathbf{P}$. According to the part "$(iii) \Rightarrow (iv)$" of the proof of Theorem 1.11, we have $\mathrm{lm}_{\mathbf{s}}(\mathbf{p}) = \mathrm{lm}_{\mathbf{d}}(\boldsymbol{\lambda})\mathrm{lm}_{\mathbf{s}}(\mathbf{P})$. Let $i \in \{1, \ldots, \rho\}$ be the index of the rightmost nonzero entry of $\mathrm{lm}_{\mathbf{d}}(\boldsymbol{\lambda})$; in particular, $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{p}) = \mathrm{rdeg}_{\mathbf{d}}(\boldsymbol{\lambda}) \geqslant s_{j_i} + \delta_i$. Furthermore, since $\mathbf{P}$ is in $\mathbf{s}$-ordered weak Popov form, $\mathrm{lm}_{\mathbf{s}}(\mathbf{P})$ has an echelon shape which implies that the rightmost nonzero element of $\mathrm{lm}_{\mathbf{s}}(\mathbf{p})$ is at index j_i. Thus $\mathbf{p}$ has $\mathbf{s}$-pivot index j_i, and since $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{p}) \geqslant s_{j_i} + \delta_i$ we obtain that its $\mathbf{s}$-pivot degree is at least δ_i.

Before moving to shifted Popov forms, we illustrate this property, insisting on the information which is brought to us by the pivots of a weak Popov form and which we do not have for reduced forms in general.

Example 1.18. Let $\mathcal{M}$ be a submodule of $\mathbb{K}[X]^m$ of rank 4, and consider the case of the uniform shift. First suppose that we know a reduced basis $\mathbf{A} \in \mathbb{K}[X]^{4 \times 4}$ of $\mathcal{M}$, whose entries have degrees as follows:

$$\mathbf{A} = \begin{bmatrix} [1] & [1] & [1] & [1] \\ [5] & [5] & [5] & [5] \\ [2] & [2] & [2] & [2] \\ [7] & [7] & [7] & [7] \end{bmatrix},$$

where $[d]$ denotes an entry of degree d. We note that the pivot entries of $\mathbf{A}$ are all located in its last column.

Then, we can assert that any nonzero vector $\mathbf{p}$ in the row space of $\mathbf{A}$, which is $\mathcal{M}$, must satisfy $\operatorname{rdeg}(\mathbf{p}) \geqslant 1 = \min(\operatorname{rdeg}(\mathbf{A}))$. However, by only considering the degrees in $\mathbf{A}$, we do not know if there may be in $\mathcal{M}$ a vector $\mathbf{q}$ whose degrees are $\begin{bmatrix}[8] & [6] & [4] & [5]\end{bmatrix}$.

Now suppose that we know a $\mathbf{0}$-ordered weak Popov basis $\mathbf{B}$ of $\mathcal{M}$, which has degrees

$$\mathbf{B} = \begin{bmatrix} [5] & [4] & [4] & [4] \\ [7] & [7] & [6] & [6] \\ [2] & [2] & [2] & [1] \\ [1] & [1] & [1] & [1] \end{bmatrix}.$$

The lemma above implies that $\mathbf{q} \notin \mathcal{M}$ since the pivot of $\mathbf{q}$ has index 2 and degree 6, which is less than the pivot degree 7 of the second row of $\mathbf{B}$. ✍

Shifted Popov forms. These are shifted ordered weak Popov forms that are further normalized through an additional constraint on the degrees in the columns containing the pivot entries. Formally, we will use the definition from [BLV99, Definition 3.3].

Definition 1.19 (Shifted Popov form)**.** *Let $\rho \leqslant m$, let $\mathbf{P} \in \mathbb{K}[X]^{\rho \times m}$ have full rank, and let $\mathbf{s} \in \mathbb{Z}^m$. Then, $\mathbf{P}$ is said to be in* $\mathbf{s}$-Popov form *if the $\mathbf{s}$-pivot indices of its rows are strictly increasing, the corresponding $\mathbf{s}$-pivot entries are monic, and in each column of $\mathbf{P}$ which contains a pivot the nonpivot entries have degree less than the pivot entry.*

Thus, a matrix in $\mathbf{s}$-Popov form is in particular in $\mathbf{s}$-ordered weak Popov form and $\mathbf{s}$-reduced. For a more detailed treatment of shifted Popov forms, the reader may refer to [BLV99, BLV06]. Shifted Popov forms are named after Popov, who introduced this form for the uniform shift in [Pop72] in the context of control theory.

Remark 1.20. The definition of $\mathbf{0}$-Popov forms in [Kai80, MS03b] slightly differs from the one above in its choice of the ordering of the rows. Instead of requiring that the $\mathbf{s}$-pivot indices be increasing as here and in [BLV99, BLV06], the definition in [Kai80, MS03b] requires that $\operatorname{rdeg}_{\mathbf{s}}(\mathbf{P})$ be non-decreasing, with increasing $\mathbf{s}$-pivot indices for rows that have identical $\mathbf{s}$-degree. ☕

For $\rho \leqslant m$, a given shift $\mathbf{s}$, and a given matrix $\mathbf{A} \in \mathbb{K}[X]^{\rho \times m}$ with full rank, there exists a unique matrix $\mathbf{P} \in \mathbb{K}[X]^{\rho \times m}$ which is in $\mathbf{s}$-Popov form and left-unimodularly equivalent to $\mathbf{A}$ [BLV99, Theorem 3.5]. We call $\mathbf{P}$ the $\mathbf{s}$-*Popov form of* $\mathbf{A}$. In terms of bases of modules, this means that for a given $\mathbb{K}[X]$-submodule $\mathcal{M} \subseteq \mathbb{K}[X]^m$ of rank ρ, there is a unique basis $\mathbf{P} \in \mathbb{K}[X]^{\rho \times m}$ of $\mathcal{M}$ which is in $\mathbf{s}$-Popov form, called the $\mathbf{s}$-*Popov basis of* $\mathcal{M}$. In particular, if we are given the descriptions of two modules and we have a procedure to compute their respective shifted Popov bases, this will directly allow us to test whether these descriptions actually represent the same module.

The *Hermite form*, defined by Hermite [Her51] in the context of triangularizing integer matrices, is a normal form for bases of submodules of $\mathbb{K}[X]^m$ that has the particularity of having a triangular shape. In fact, Hermite forms coincide with shifted Popov forms for specific shifts that make them triangular, as detailed in [BLV06, Lemma 2.6]. We give more details about Hermite forms in Section 3.2.

For example, the matrix $\mathbf{P}$ in Example 1.12 is in Hermite form, and it is also in $(0,3)$-Popov form. On the other hand, the matrix $\mathbf{A}$ in the same example is in $\mathbf{0}$-ordered weak Popov form, but not in $\mathbf{0}$-Popov form since the entry at position $(2,1)$ has the same degree as the pivot in position $(1,1)$; $\mathbf{A}$ may be transformed into its $\mathbf{0}$-Popov form by a constant elementary row operation.

1.2 Designing fast algorithms for shifted Popov bases

In this section, we discuss general ingredients and bounds that we will use in our algorithms for computing shifted Popov bases. Here, we focus on the case of submodules of $\mathbb{K}[X]^m$ of rank m, whose bases are thus square nonsingular polynomial matrices.

We note that most of Section 1.2.1 can be extended to submodules of rank less than m and to rectangular bases; however, this generality is not needed for our algorithms and would thus make the presentation unnecessarily technical.

1.2.1 Finding and using the minimal degree

Here, we study the shifted minimal degree of a submodule $\mathcal{M} \subseteq \mathbb{K}[X]^m$ of rank m. After giving some basic properties, we discuss how it can be used to compute shifted Popov forms and how it can be found by a divide-and-conquer approach.

Minimal degree. A basis of such a module $\mathcal{M}$ is represented by *square*, nonsingular matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$. Then, $\mathbf{P}$ is in $\mathbf{s}$-ordered weak Popov form if and only if its $\mathbf{s}$-pivot entries are on the diagonal. In this case, the $\mathbf{s}$-pivot degree $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$ of $\mathbf{P}$ is the tuple of the degrees of its diagonal entries, and we have $\operatorname{rdeg}_{\mathbf{s}}(\mathbf{P}) = \mathbf{s}+\boldsymbol{\delta}$ as well as $\deg(\det(\mathbf{P})) = |\boldsymbol{\delta}|$. If $\mathbf{P}$ is furthermore in $\mathbf{s}$-Popov form, then its $\mathbf{s}$-pivot degree coincides with its column degree: $\boldsymbol{\delta} = \operatorname{cdeg}(\mathbf{P})$. We have the following straightforward characterizations in terms of shifted leading matrices.

Lemma 1.21. *Let $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ and $\mathbf{s} \in \mathbb{Z}^m$. Then,*

- *$\mathbf{P}$ is in $\mathbf{s}$-ordered weak Popov form if and only if its $\mathbf{s}$-leading matrix is lower triangular and invertible;*
- *$\mathbf{P}$ is in $\mathbf{s}$-Popov form if and only if its $\mathbf{s}$-leading matrix is unit lower triangular and the $\mathbf{0}$-leading matrix of $\mathbf{P}^\mathsf{T}$ is the identity matrix.*

Note that the second item was chosen as the definition of shifted Popov forms in [BLV99, Definition 3.3]. We are particularly interested in the following degrees.

Definition 1.22 (Minimal degree)**.** *For a $\mathbb{K}[X]$-submodule $\mathcal{M} \subseteq \mathbb{K}[X]^m$ of rank m and any $\mathbf{s} \in \mathbb{Z}^m$, the* $\mathbf{s}$-minimal degree of $\mathcal{M}$ *is the $\mathbf{s}$-pivot degree of the $\mathbf{s}$-Popov basis of $\mathcal{M}$.*

Let us denote by $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m)$ the $\mathbf{s}$-minimal degree of $\mathcal{M}$. Then, the determinant of any basis of $\mathcal{M}$ has degree $|\boldsymbol{\delta}| = \delta_1 + \cdots + \delta_m$. Besides, according to Lemma 1.17, $\boldsymbol{\delta}$ also

gives useful information concerning the degrees of elements of $\mathcal{M}$: if a nonzero $\mathbf{p} \in \mathcal{M}$ has $\mathbf{s}$-pivot index $j \in \{1, \ldots, m\}$, then the $\mathbf{s}$-pivot degree of $\mathbf{p}$ must be at least δ_j, or in other words, we must have $\operatorname{rdeg}_{\mathbf{s}}(\mathbf{p}) \geqslant s_j + \delta_j$. We now illustrate this notion with an example, showing in particular that the minimal degree cannot be deduced from the row degrees of a reduced basis in general.

Example 1.23. As in Example 1.18, let us consider a module $\mathcal{M}$ which admits a $\mathbf{0}$-reduced matrix with degrees

$$\mathbf{A} = \begin{bmatrix} [1] & [1] & [1] & [1] \\ [5] & [5] & [5] & [5] \\ [2] & [2] & [2] & [2] \\ [7] & [7] & [7] & [7] \end{bmatrix}.$$

We know that the $\mathbf{0}$-Popov form of $\mathbf{A}$ has the same row degrees as $\mathbf{A}$, up to permutation. Besides, for the uniform shift the pivot degree of a row vector coincides with its $\mathbf{0}$-degree; hence the minimal degree of $\mathcal{M}$ is a permutation of $(1, 5, 2, 7)$. Yet, without additional information, we do not know precisely which permutation it is.

More generally, if we have an $\mathbf{s}$-reduced basis $\mathbf{A} \in \mathbb{K}[X]^{m \times m}$ of $\mathcal{M} \subseteq \mathbb{K}[X]^m$ for an arbitrary shift $\mathbf{s}$, then the shifted minimal degree of $\mathcal{M}$ will be $\mathbf{d} - \mathbf{s}$, where $\mathbf{d}$ is some permutation of $\operatorname{rdeg}_{\mathbf{s}}(\mathbf{A})$ which is unknown a priori. ✍

The next lemma states that any row vector $\mathbf{p} \in \mathbb{K}[X]^{1 \times m}$ can be written uniquely as the sum of an element of $\mathcal{M}$ and a *remainder* $\mathbf{r} = [r_1, \ldots, r_m]$ such that $\deg(r_j) < \delta_j$ for $1 \leqslant j \leqslant m$. In other words, the $\mathbf{s}$-minimal degree gives the structure of the quotient module $\mathbb{K}[X]^m/\mathcal{M}$: it is isomorphic to $\mathbb{K}[X]/\langle X^{\delta_1} \rangle \times \cdots \times \mathbb{K}[X]/\langle X^{\delta_m} \rangle$; that is, it admits the monomial basis $\cup_{1 \leqslant j \leqslant m} \{X^d \mathbf{c}_j, 0 \leqslant d < \delta_j\}$ where $\mathbf{c}_j \in \mathbb{K}[X]^{1 \times m}$ is the coordinate vector with 1 at index j.

Lemma 1.24 (Division with remainder)**.** *Let $\mathbf{s} \in \mathbb{Z}^m$, let $\mathbf{P} \in \mathbb{K}[X]^{m \times m}$ be in $\mathbf{s}$-Popov form, and let $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$ be the $\mathbf{s}$-pivot degree of $\mathbf{P}$. Then, for any row vector $\mathbf{p} \in \mathbb{K}[X]^{1 \times m}$, there exists a unique $(\mathbf{q}, \mathbf{r}) \in \mathbb{K}[X]^{1 \times m} \times \mathbb{K}[X]^{1 \times m}$ such that $\mathbf{p} = \mathbf{q}\mathbf{P} + \mathbf{r}$ and $\operatorname{rdeg}_{-\boldsymbol{\delta}}(\mathbf{r}) < 0$.*

For this result, the reader may refer to [Kai80, Theorem 6.3-15], noting that a matrix in $\mathbf{s}$-Popov form is in particular $\mathbf{0}$-*column* reduced.

By definition, left-unimodularly equivalent $\mathbf{s}$-reduced matrices have the same $\mathbf{s}$-row degree up to permutation. In the case of shifted weak Popov forms, knowing the pivot indices leads us to a more precise statement. Namely, the next result states that the $\mathbf{s}$-row degree is invariant among left-unimodularly equivalent matrices in $\mathbf{s}$-ordered weak Popov form. This implies that left-unimodularly equivalent matrices in $\mathbf{s}$-ordered weak Popov form have the same $\mathbf{s}$-pivot degree, which must then be the $\mathbf{s}$-minimal degree $\boldsymbol{\delta}$ of the module they generate.

In short, the next lemma shows that $\boldsymbol{\delta}$ can be read off from any basis of $\mathcal{M}$ in $\mathbf{s}$-weak Popov form; we recall from Example 1.23 that this is not true concerning $\mathbf{s}$-reduced bases.

Lemma 1.25. *Let $\mathbf{s} \in \mathbb{Z}^m$ and let $\mathbf{P}$ and $\mathbf{Q}$ in $\mathbb{K}[X]^{m \times m}$ be two left-unimodularly equivalent matrices in $\mathbf{s}$-ordered weak Popov form. Then $\mathbf{P}$ and $\mathbf{Q}$ have the same $\mathbf{s}$-row degree.*

As a corollary, for any submodule $\mathcal{M} \subseteq \mathbb{K}[X]^m$ of rank m, if $\mathbf{P} \in \mathbb{K}[X]^{m \times m}$ is an $\mathbf{s}$-weak Popov basis of $\mathcal{M}$ with $\mathbf{s}$-pivot index $(j_1, \ldots, j_m)$ and $\mathbf{s}$-pivot degree $(\delta_1, \ldots, \delta_m)$,

and $(\pi_1, \ldots, \pi_m)$ *is the permutation of* $\{1, \ldots, m\}$ *such that* $(j_{\pi_1}, \ldots, j_{\pi_m}) = (1, 2, \ldots, m)$*, then the* $\mathbf{s}$*-minimal degree of* $\mathcal{M}$ *is* $(\delta_{\pi_1}, \ldots, \delta_{\pi_m})$*.*

Proof. Since $\mathbf{P}$ and $\mathbf{Q}$ are in $\mathbf{s}$-ordered weak Popov form, their respective $\mathbf{s}$-pivot degrees are $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}) - \mathbf{s}$ and $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{Q}) - \mathbf{s}$. Then, since $\mathbf{P}_{i,*}$ is in the row space of $\mathbf{Q}$ for $1 \leqslant i \leqslant m$, Lemma 1.17 implies that $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}) - \mathbf{s} \leqslant \mathrm{rdeg}_{\mathbf{s}}(\mathbf{Q}) - \mathbf{s}$. The same argument applied to the rows of $\mathbf{Q}$ leads to the equality $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}) - \mathbf{s} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{Q}) - \mathbf{s}$, hence $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}) = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{Q})$.

The corollary follows, since $(\delta_{\pi_1}, \ldots, \delta_{\pi_m})$ is the $\mathbf{s}$-pivot degree of the matrix $\mathbf{P}$ with rows permuted so as to obtain an $\mathbf{s}$-ordered weak Popov form; and then its $\mathbf{s}$-Popov form must have the same $\mathbf{s}$-pivot degree since it is in particular in $\mathbf{s}$-ordered weak Popov form. ☙

Using the minimal degree. We explain now explicitly how we can use the knowledge of the shifted minimal degree in the computation of shifted Popov bases. The central tool is the next lemma, which is an extension of [SS11, Lemmas 15 and 17] to the case of any shift $\mathbf{s}$. It states that, as soon as the $\mathbf{s}$-minimal degree $\boldsymbol{\delta}$ of the module is known, the computation of the $\mathbf{s}$-Popov basis $\mathbf{P}$ essentially boils down to the computation of a $-\boldsymbol{\delta}$-reduced basis $\mathbf{R}$. Furthermore, as we will detail in Section 1.2.2, the shift $-\boldsymbol{\delta}$ is among those for which it is reasonable to hope that $\mathbf{R}$ can be computed efficiently.

Lemma 1.26. *Let* $\mathbf{s} \in \mathbb{Z}^m$ *and let* $\mathbf{P} \in \mathbb{K}[X]^{m \times m}$ *be in* $\mathbf{s}$*-Popov form with* $\mathbf{s}$*-pivot degree* $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$*. Then* $\mathbf{P}$ *is also in* $-\boldsymbol{\delta}$*-Popov form, and we have* $\mathrm{rdeg}_{-\boldsymbol{\delta}}(\mathbf{P}) = \mathbf{0}$*. In particular, for any matrix* $\mathbf{R} \in \mathbb{K}[X]^{m \times m}$ *which is left-unimodularly equivalent to* $\mathbf{P}$ *and* $-\boldsymbol{\delta}$*-reduced,* $\mathbf{R}$ *has column degree* $\boldsymbol{\delta}$*, and* $\mathbf{P} = \mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})^{-1}\mathbf{R}$*.*

Proof. Since $\mathbf{P}$ is in $\mathbf{s}$-Popov form we already have $\mathrm{lm}_{\mathbf{0}}(\mathbf{P}^{\mathsf{T}}) = \mathbf{I}_m$, and then according to Lemma 1.21 it remains to prove that $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{P})$ is unit lower triangular. In fact, considering $\mathbf{P}\mathbf{X}^{-\boldsymbol{\delta}}$ which has only coefficients of negative degree except for those of degree zero which form the identity matrix, we obtain that $\mathrm{rdeg}_{-\boldsymbol{\delta}}(\mathbf{P}) = \mathbf{0}$ and $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{P}) = \mathbf{I}_m$.

Now, let $\mathbf{R}$ be a $-\boldsymbol{\delta}$-reduced matrix left-unimodularly equivalent to $\mathbf{P}$. Then, we have $\mathrm{rdeg}_{-\boldsymbol{\delta}}(\mathbf{R}) = \mathrm{rdeg}_{-\boldsymbol{\delta}}(\mathbf{P}) = \mathbf{0}$, so that we can write $\mathbf{R} = \mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})\mathbf{X}^{\boldsymbol{\delta}} + \mathbf{Q}$ with the j-th column of $\mathbf{Q}$ of degree less than δ_j, where $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m)$. In particular, since $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})$ is invertible, this yields $\mathrm{cdeg}(\mathbf{R}) = \boldsymbol{\delta}$. Besides, we obtain $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})^{-1}\mathbf{R} = \mathbf{X}^{\boldsymbol{\delta}} + \mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})^{-1}\mathbf{Q}$, and the j-th column of $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})^{-1}\mathbf{Q}$ has degree less than δ_j. Thus $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})^{-1}\mathbf{R}$ is in $-\boldsymbol{\delta}$-Popov form and unimodularly equivalent to $\mathbf{P}$, hence equal to $\mathbf{P}$. ☙

Yet, for the knowledge of the $\mathbf{s}$-minimal degree $\boldsymbol{\delta}$ to be useful in algorithms, one has to be able to compute it without computing the $\mathbf{s}$-Popov basis itself. Such a nice situation, which we will focus on in Section 3.2.3 and Chapter 16, is that of the computation of the Hermite form of $\mathcal{M}$ when it is given by some basis $\mathbf{A} \in \mathbb{K}[X]^{m \times m}$. In this case, two efficient algorithms have been given to compute the diagonal entries of this Hermite form [GS11, Zho12]. These entries give in particular the minimal degree $\boldsymbol{\delta}$; we note that, although the polynomials themselves are known, [GS11, Zho12] only use their degrees $\boldsymbol{\delta}$ to complete the computation of the whole Hermite form.

Concerning our main problems, the module is described by equations like in the examples of Hermite-Padé approximation or bivariate interpolation that we have given above.

In this context, we show now that the shifted minimal degree behaves well with algorithms based on a divide-and-conquer approach, and thus that it can be found without computing the full shifted Popov basis.

Finding the minimal degree. Placing ourselves in an abstract setting, suppose that we have some description of a module $\mathcal{M} \subseteq \mathbb{K}[X]^m$ of rank m and we want to compute a basis $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ of $\mathcal{M}$. Depending on the context, $\mathbf{P}$ should be in shifted reduced form or in shifted Popov form; we will study both situations.

When the description of $\mathcal{M}$ allows it, we split the problem into two subproblems, as follows. The first subproblem asks to compute a basis $\mathbf{P}^{(1)} \in \mathbb{K}[X]^{m\times m}$ of a module $\mathcal{M}^{(1)}$ such that $\mathcal{M} \subseteq \mathcal{M}^{(1)} \subseteq \mathbb{K}[X]^m$; remark that these inclusions imply that the rank of $\mathcal{M}^{(1)}$ is m. Then, we wish to use the knowledge of $\mathbf{P}^{(1)}$ to progress towards obtaining a basis of $\mathcal{M}$. For this, we express the elements of $\mathcal{M} \subseteq \mathcal{M}^{(1)}$ on the basis $\mathbf{P}^{(1)}$:

$$\begin{aligned}\mathcal{M} &= \{\mathbf{p} \in \mathcal{M}^{(1)} \mid \mathbf{p} \in \mathcal{M}\}\\ &= \{\boldsymbol{\lambda}\mathbf{P}^{(1)}, \boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m} \mid \boldsymbol{\lambda}\mathbf{P}^{(1)} \in \mathcal{M}\} = \mathcal{M}^{(2)}\mathbf{P}^{(1)},\end{aligned}$$

where $\mathcal{M}^{(2)} = \{\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m} \mid \boldsymbol{\lambda}\mathbf{P}^{(1)} \in \mathcal{M}\}$. Then, the second subproblem is to compute a basis $\mathbf{P}^{(2)}$ of $\mathcal{M}^{(2)}$.

Example 1.27 (Hermite-Padé approximation). Following on from Example 1.5, let $\mathbf{F} \in \mathbb{K}[X]^{m\times 1}$ be a vector, let D be an even positive integer, and consider

$$\mathcal{M} = \{\mathbf{p} \in \mathbb{K}[X]^{1\times m} \ \mid\ \mathbf{pF} = \mathbf{0} \bmod X^D\},$$

which is of rank m. Then, one may consider the module

$$\mathcal{M}^{(1)} = \{\mathbf{p} \in \mathbb{K}[X]^{1\times m} \ \mid\ \mathbf{pF} = \mathbf{0} \bmod X^{D/2}\},$$

which is such that $\mathcal{M} \subseteq \mathcal{M}^{(1)} \subseteq \mathbb{K}[X]^m$. As a first subproblem, we compute a basis $\mathbf{P}^{(1)}$ of $\mathcal{M}^{(1)}$, which is an instance of Hermite-Padé approximation at order $D/2$. Then, for any $\mathbf{p} \in \mathbb{K}[X]^{1\times m}$,

$$\begin{aligned}\mathbf{p} \in \mathcal{M} \ &\Leftrightarrow\ \mathbf{p} \in \mathcal{M}^{(1)} \ \text{ and } \ \mathbf{pF} = 0 \bmod X^D\\ &\Leftrightarrow\ \mathbf{p} = \boldsymbol{\lambda}\mathbf{P}^{(1)} \text{ for some } \boldsymbol{\lambda} \text{ such that } \boldsymbol{\lambda}\mathbf{P}^{(1)}\mathbf{F} = 0 \bmod X^D\\ &\Leftrightarrow\ \mathbf{p} = \boldsymbol{\lambda}\mathbf{P}^{(1)} \text{ for some } \boldsymbol{\lambda} \text{ such that } \boldsymbol{\lambda}\mathbf{G} = 0 \bmod X^{D/2},\end{aligned}$$

where $\mathbf{G} = (X^{-D/2}\mathbf{P}^{(1)}\mathbf{F}) \bmod X^{D/2}$ is often called the *residual* in this context. With the notation above, we have

$$\mathcal{M}^{(2)} = \{\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m} \ \mid\ \boldsymbol{\lambda}\mathbf{G} = \mathbf{0} \bmod X^{D/2}\},$$

and thus our second subproblem is to compute a basis $\mathbf{P}^{(2)}$ of $\mathcal{M}^{(2)}$, which is an instance of Hermite-Padé approximation at order $D/2$ for the vector $\mathbf{G}$. ✍

The computed bases $\mathbf{P}^{(1)}$ and $\mathbf{P}^{(2)}$ are combined into the product $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$, which is a basis of $\mathcal{M}$ as we prove below. Most importantly, computing an $\mathbf{s}$-reduced basis is consistent with this strategy, via the computation of $\mathbf{P}^{(1)}$ in $\mathbf{s}$-reduced form and of $\mathbf{P}^{(2)}$ in $\mathbf{t}$-reduced form for a well-chosen shift $\mathbf{t}$. These properties, stated in items (i) and (ii) of the theorem below, are behind most algorithms for Hermite-Padé approximation and similar problems, including the iterative ones of [VBB92, BL94, BL00] and the divide-and-conquer ones of [BL94, GJV03, ZL12, GL14].

Theorem 1.28. *Let $\mathcal{M} \subseteq \mathcal{M}^{(1)}$ be two $\mathbb{K}[X]$-submodules of $\mathbb{K}[X]^m$ of rank m, and let $\mathbf{P}^{(1)} \in \mathbb{K}[X]^{m\times m}$ be a basis of $\mathcal{M}^{(1)}$. Let further $\mathbf{s} \in \mathbb{Z}^m$ and $\mathbf{t} = \operatorname{rdeg}_{\mathbf{s}}(\mathbf{P}^{(1)})$. Then,*

(i) the rank of the module $\mathcal{M}^{(2)} = \{\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m} \mid \boldsymbol{\lambda}\mathbf{P}^{(1)} \in \mathcal{M}\}$ is m, and for any basis $\mathbf{P}^{(2)} \in \mathbb{K}[X]^{m\times m}$ of $\mathcal{M}^{(2)}$, the product $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$ is a basis of $\mathcal{M}$;

(ii) if $\mathbf{P}^{(1)}$ is $\mathbf{s}$-reduced and $\mathbf{P}^{(2)}$ is $\mathbf{t}$-reduced, then $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$ is $\mathbf{s}$-reduced;

(iii) if $\mathbf{P}^{(1)}$ is in $\mathbf{s}$-ordered weak Popov form and $\mathbf{P}^{(2)}$ is in $\mathbf{t}$-ordered weak Popov form, then $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$ is in $\mathbf{s}$-ordered weak Popov form;

(iv) if $\boldsymbol{\delta}^{(1)}$ is the $\mathbf{s}$-minimal degree of $\mathcal{M}^{(1)}$ and $\boldsymbol{\delta}^{(2)}$ is the $\mathbf{t}$-minimal degree of $\mathcal{M}^{(2)}$, then the $\mathbf{s}$-minimal degree of $\mathcal{M}$ is $\boldsymbol{\delta}^{(1)} + \boldsymbol{\delta}^{(2)}$.

Proof. (i) To prove that $\mathcal{M}^{(2)}$ has rank m, we consider the adjugate $\operatorname{adj}(\mathbf{P}^{(1)}) \in \mathbb{K}[X]^{m\times m}$ of $\mathbf{P}^{(1)}$, that is, the transpose of its cofactor matrix. Then, $\operatorname{adj}(\mathbf{P}^{(1)})\mathbf{P}^{(1)} = \det(\mathbf{P}^{(1)})\mathbf{I}_m$ and thus for any $\mathbf{p} \in \mathcal{M}$ we have $\mathbf{p}\operatorname{adj}(\mathbf{P}^{(1)})\mathbf{P}^{(1)} = \det(\mathbf{P}^{(1)})\mathbf{p} \in \mathcal{M}$. Hence $\mathcal{M}\operatorname{adj}(\mathbf{P}^{(1)}) \subseteq \mathcal{M}^{(2)}$ and it remains to show that $\mathcal{M}\operatorname{adj}(\mathbf{P}^{(1)})$ has rank m, which follows from the non-singularity of $\operatorname{adj}(\mathbf{P}^{(1)})$.

The rows of $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$ are $\mathbb{K}[X]$-linearly independent since its determinant is nonzero. Now let $\mathbf{p} \in \mathcal{M}$; we want to prove that $\mathbf{p}$ is a $\mathbb{K}[X]$-linear combination of the rows of $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$. First, $\mathbf{p} \in \mathcal{M}^{(1)}$, so there exists $\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m}$ such that $\mathbf{p} = \boldsymbol{\lambda}\mathbf{P}^{(1)}$. But then $\boldsymbol{\lambda} \in \mathcal{M}^{(2)}$, and thus there exists $\boldsymbol{\mu} \in \mathbb{K}[X]^{1\times m}$ such that $\boldsymbol{\lambda} = \boldsymbol{\mu}\mathbf{P}^{(2)}$. This yields the combination $\mathbf{p} = \boldsymbol{\mu}\mathbf{P}^{(2)}\mathbf{P}^{(1)}$.

(ii) Let $\mathbf{d} = \operatorname{rdeg}_{\mathbf{t}}(\mathbf{P}^{(2)})$; by the predictable-degree property from Theorem 1.11, we have $\mathbf{d} = \operatorname{rdeg}_{\mathbf{s}}(\mathbf{P}^{(2)}\mathbf{P}^{(1)})$. Using $\mathbf{X}^{-\mathbf{d}}\mathbf{P}^{(2)}\mathbf{P}^{(1)}\mathbf{X}^{\mathbf{s}} = \mathbf{X}^{-\mathbf{d}}\mathbf{P}^{(2)}\mathbf{X}^{\mathbf{t}}\mathbf{X}^{-\mathbf{t}}\mathbf{P}^{(1)}\mathbf{X}^{\mathbf{s}}$, we obtain that $\operatorname{lm}_{\mathbf{s}}(\mathbf{P}^{(2)}\mathbf{P}^{(1)}) = \operatorname{lm}_{\mathbf{t}}(\mathbf{P}^{(2)})\operatorname{lm}_{\mathbf{s}}(\mathbf{P}^{(1)})$. According to Theorem 1.11, our assumption implies that $\operatorname{lm}_{\mathbf{t}}(\mathbf{P}^{(2)})$ and $\operatorname{lm}_{\mathbf{s}}(\mathbf{P}^{(1)})$ are invertible. Therefore $\operatorname{lm}_{\mathbf{s}}(\mathbf{P}^{(2)}\mathbf{P}^{(1)})$ is invertible as well, and $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$ is $\mathbf{s}$-reduced.

(iii) Using Lemma 1.21, the result follows since $\operatorname{lm}_{\mathbf{s}}(\mathbf{P}^{(2)}\mathbf{P}^{(1)}) = \operatorname{lm}_{\mathbf{t}}(\mathbf{P}^{(2)})\operatorname{lm}_{\mathbf{s}}(\mathbf{P}^{(1)})$ is lower triangular and invertible.

(iv) Let $\mathbf{P}^{(1)}$ be the $\mathbf{s}$-Popov basis of $\mathcal{M}^{(1)}$ and $\mathbf{P}^{(2)}$ be the $\mathbf{t}$-Popov basis of $\mathcal{M}^{(2)}$. Then, by the item (iii) above, $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$ is in $\mathbf{s}$-ordered weak Popov form. Thanks to Lemma 1.25, it remains to show that the $\mathbf{s}$-pivot degree of $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$ is $\boldsymbol{\delta}^{(1)} + \boldsymbol{\delta}^{(2)}$. Since $\mathbf{P}^{(1)}$ and $\mathbf{P}^{(2)}$ are in $\mathbf{s}$- and $\mathbf{t}$-Popov form, we have $\mathbf{t} = \operatorname{rdeg}_{\mathbf{s}}(\mathbf{P}^{(1)}) = \mathbf{s} + \boldsymbol{\delta}^{(1)}$ and $\operatorname{rdeg}_{\mathbf{t}}(\mathbf{P}^{(2)}) = \mathbf{t} + \boldsymbol{\delta}^{(2)}$; the conclusion then follows from $\operatorname{rdeg}_{\mathbf{s}}(\mathbf{P}^{(2)}\mathbf{P}^{(1)}) = \operatorname{rdeg}_{\mathbf{t}}(\mathbf{P}^{(2)}) = \mathbf{t} + \boldsymbol{\delta}^{(2)} = \mathbf{s} + \boldsymbol{\delta}^{(1)} + \boldsymbol{\delta}^{(2)}$. ∎

We remark however that for $\mathbf{P}^{(1)}$ in $\mathbf{s}$-Popov form and $\mathbf{P}^{(2)}$ in $\mathbf{t}$-Popov form, the product $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$ is generally not in $\mathbf{s}$-Popov form. One may want to first compute $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$

and then normalize it into $\mathbf{s}$-Popov form; while this will give the sought shifted Popov form, this strategy is too costly in the worst cases. In fact, even the number of coefficients from $\mathbb{K}$ needed to represent the product $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$ may be beyond our target cost, as we will detail in Section 1.2.2.

To circumvent this difficulty, the algorithms presented in Sections 2.3 to 2.5 avoid to directly follow this approach based on polynomial matrix multiplication. Instead, they rely on the item (iv) above. It implies that, if we have obtained recursively $\mathbf{P}^{(1)}$ and $\mathbf{P}^{(2)}$ in $\mathbf{s}$- and $\mathbf{t}$-Popov form, then the $\mathbf{s}$-minimal degree $\boldsymbol{\delta}$ of $\mathcal{M}$ can be simply obtained by adding the tuple of diagonal degrees of $\mathbf{P}^{(1)}$ and $\mathbf{P}^{(2)}$. Then, we rely in particular on Lemma 1.26 to exploit the knowledge of $\boldsymbol{\delta}$ in order to efficiently compute the $\mathbf{s}$-Popov basis of $\mathcal{M}$.

1.2.2 Size of bases and target costs

In this section, we discuss bounds on the size of the bases of a module, where by size, we mean the number of field elements used to represent them. In this thesis, except when indicated, we will always assume a dense representation of polynomial matrices, meaning that the size of a matrix $\mathbf{P} = [p_{ij}]_{i,j} \in \mathbb{K}[X]^{m\times n}$ is measured as $mn + \sum_{\{(i,j)\,|\,p_{ij}\neq 0\}} \deg(p_{ij})$. Studying the size of the matrices we compute is important, since it provides trivial lower bounds: it is expected that an algorithm which returns a basis of size S will have to perform some computation for each of the output field element, and thus it will use $\Omega(S)$ operations in $\mathbb{K}$ assuming that no structure of the basis is exploited. These size bounds can also be used to set up target costs for our algorithms.

Here we only consider modules $\mathcal{M} \subseteq \mathbb{K}[X]^m$ of rank m, which are thus represented by a square, nonsingular polynomial matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$. If we do not make any specific requirement on $\mathbf{P}$, it may have entries of arbitrary large degree. For example, the bases of the module $\mathbb{K}[X]^{1\times m}$ form the set of unimodular matrices, which contains all unit upper or lower triangular matrices (see Example 1.2). On the other hand, as soon as we require some minimality, the degree of the determinant of the computed bases will play a central role in their size.

Shifted reduced bases and average row degree. The constraints of being a reduced form immediately restrict the possibilities in terms of degrees. For example, the identity matrix is a basis of $\mathbb{K}[X]^{1\times m}$ which is $\mathbf{0}$-reduced, and thus any other $\mathbf{0}$-reduced basis must have the same $\mathbf{0}$-row degree, or in other words, must be a nonsingular *constant* matrix. A convenient tool for analyzing the degrees of shifted reduced matrices is provided by the determinant, thanks to Theorem 1.11: for instance, any unimodular matrix $\mathbf{U}$ which is $\mathbf{0}$-reduced satisfies $|\mathrm{rdeg}(\mathbf{U})| = \deg(\det(\mathbf{U})) = 0$.

The shift $\mathbf{s} \in \mathbb{Z}^m$, as a set of degree weights on the columns of $\mathbf{s}$-reduced bases, affects how the degrees of the entries of such bases are distributed. While in our problems the precise degree distribution in the ouput is unknown a priori, we still have a global control on it, as follows. We will *always* have, directly from the input, a bound $D \in \mathbb{Z}_{\geqslant 0}$ on the degree of the determinant of any basis of $\mathcal{M}$. Furthermore, we will have a related bound on the size of the input, which will often be $\mathcal{O}(mD)$, or in some cases $\mathcal{O}(m^2\lceil D/m\rceil)$.

Then, this quantity D provides us with a bound on the sum of the shifted row degrees of any $\mathbf{s}$-reduced basis $\mathbf{P}$ of $\mathcal{M}$:

$$|\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})| = \deg(\det(\mathbf{P})) + |\mathbf{s}| \leqslant D + |\mathbf{s}|.$$

After ensuring that the shift has nonnegative entries, by considering $\mathbf{s} - \min(\mathbf{s})$ instead of $\mathbf{s}$ without loss of generality, we bound the sum of actual row degrees of $\mathbf{P}$ as

$$|\mathrm{rdeg}(\mathbf{P})| \leqslant |\mathrm{rdeg}_{\mathbf{s}-\min(\mathbf{s})}(\mathbf{P})| \leqslant D + \xi,$$

where $\xi = |\mathbf{s} - \min(\mathbf{s})|$. This states that the average row degree of $\mathbf{P}$ is at most $(D+\xi)/m$. In particular, $\mathbf{P}$ can be represented using at most $m^2 + m(D+\xi)$ field elements.

Our algorithms will fundamentally rely on polynomial matrix multiplication, as for example in the general divide-and-conquer scheme presented in Section 1.2.1, where one computes two bases and multiply them together. Having the above bound, we expect to multiply matrices of dimensions $m \times m$ and average row degree in $\mathcal{O}((D+\xi)/m)$, with the product also having average row degree $\mathcal{O}((D+\xi)/m)$. There are algorithms specialized to this kind of situations, and thus a reasonable target cost is $\tilde{\mathcal{O}}(m^\omega + m^{\omega-1}(D+\xi)) = \tilde{\mathcal{O}}(m^\omega \lceil (D+\xi)/m \rceil)$, where the ceiling function takes into account the possibility that $D+\xi$ be negligible with respect to m.

To see why shifted reduced matrices may still have entries of large and unbalanced degrees, here is an example of a unimodular matrix $\mathbf{U}$ in $\mathbf{s}$-ordered weak Popov form and such that $\mathrm{rdeg}(\mathbf{U}) = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{U}) = \mathbf{s}$, and $|\mathrm{rdeg}(\mathbf{U})| = |\mathrm{rdeg}_{\mathbf{s}}(\mathbf{U})| = \deg(\det(\mathbf{U})) + \xi$. Assuming that $\mathbf{s}$ is non-decreasing and $\min(\mathbf{s}) = s_1 = 0$, we consider

$$\mathbf{U} = \begin{bmatrix} 1 & & & \\ [s_2] & 1 & & \\ \vdots & & \ddots & \\ [s_m] & & & 1 \end{bmatrix},$$

where $[d]$ denotes an entry of degree d. The size of this matrix, in terms of number of field elements, is not even the worst-case $\Theta(m^2+m(D+\xi))$, which can only be reached for some specific shifts. Yet, currently known fast polynomial matrix multiplication algorithms do not take size into account, but only the average row degree, for which this particular matrix achieves the worst case $\deg(\det(\cdot)) + \xi$.

Shifted Popov forms and average column degree. Concerning shifted Popov forms, the situation is simpler since the bound D directly gives control on their average *column* degree. Indeed, for a matrix $\mathbf{P}$ in $\mathbf{s}$-Popov form, the column degree of $\mathbf{P}$ coincides with its $\mathbf{s}$-pivot degree, that is, $\mathrm{cdeg}(\mathbf{P}) = \boldsymbol{\delta}$. On the other hand, since $\mathbf{P}$ is also column reduced, we have $|\mathrm{cdeg}(\mathbf{P})| = \deg(\det(\mathbf{P}))$. Then, having $\deg(\det(\mathbf{P})) \leqslant D$, the sum of column degrees satisfies $|\mathrm{cdeg}(\mathbf{P})| = |\boldsymbol{\delta}| = \deg(\det(\mathbf{P})) \leqslant D$, and the average column degree of $\mathbf{P}$ is at most D/m. Thus, we readily have that the size of $\mathbf{P}$ is in $\mathcal{O}(m^2+mD) = \mathcal{O}(m^2\lceil D/m \rceil)$.

In particular, in our problems where the size of the input is within the same bound, this means that one may hope for algorithms with cost bound $\tilde{\mathcal{O}}(m^\omega \lceil D/m \rceil)$. This is smaller in general than the target cost given above for shifted reduced forms, which was

dependent on $\xi = |\mathbf{s} - \min(\mathbf{s})|$. Yet, as remarked in Section 1.2.1, shifted Popov forms do not behave well regarding matrix multiplication, so that additional techniques have to be used in order to compute shifted Popov bases efficiently. To recall Section 1.2.1, currently known fast algorithms for computing shifted Popov bases focus on finding the minimal degree $\boldsymbol{\delta}$, and then use $\boldsymbol{\delta}$ to reduce the problem to the computation of a shifted reduced form for a shift that is "not far from uniform". We now discuss this case in more detail.

Assumptions on the shift. These are designed to both hold in many interesting situations and ensure that the computed shifted reduced bases have a size that remains within the size $\mathcal{O}(m^2\lceil D/m\rceil)$ of the input. This suggests a target cost of the form $\mathcal{O}\tilde{}(m^\omega\lceil D/m\rceil)$, while according to the discussion above, such a cost seems infeasible for the computation of $\mathbf{s}$-reduced bases in general, where $\xi = |\mathbf{s} - \min(\mathbf{s})|$ can be significantly larger than D.

From the degree bounds above, the assumption

$$\mathcal{H}_{\mathbf{s},1}: \quad \xi = |\mathbf{s} - \min(\mathbf{s})| \in \mathcal{O}(D), \tag{1.2}$$

ensures that an $\mathbf{s}$-reduced matrix $\mathbf{P}$ with $\deg(\det(\mathbf{P})) \leqslant D$ satisfies $|\mathrm{rdeg}(\mathbf{P})| \in \mathcal{O}(D)$. This assumption states that $\mathbf{s}$ is not far from uniform, around its minimum value.

The second commonly encountered assumption is linked to the encoding of degree constraints into shifts as in Remark 1.9. In this context, one has some prescribed bounds $N_1, \ldots, N_m \in \mathbb{Z}_{>0}$ on the output polynomial vectors, and assumes $N_1 + \cdots + N_m = D + 1$ (for example in Hermite-Padé approximation [Her93, Pad94]), or more generally that the sum $N_1 + \cdots + N_m$ is greater than D but still in $\mathcal{O}(D)$ (for example in the bivariate interpolation step of the Guruswami-Sudan algorithm [Sud97, GS99]).

In terms of the negative shift $\mathbf{s} = (-N_1, \ldots, -N_m)$, we have $|-\mathbf{s}| = N_1 + \cdots + N_m$; the corresponding assumption that we encounter for example in [ZL12] is the following.

$$\mathcal{H}_{\mathbf{s},2}: \quad |\max(\mathbf{s}) - \mathbf{s}| \in \mathcal{O}(D). \tag{1.3}$$

This assumption states that $\mathbf{s}$ is not far from uniform, around its maximum value. As we prove in the next paragraph, $\mathcal{H}_{\mathbf{s},2}$ implies that the sum of the degrees in an $\mathbf{s}$-reduced matrix $\mathbf{P}$ with $\deg(\det(\mathbf{P})) \leqslant D$ is in $\mathcal{O}(mD)$; or, in other words, the average degree of $\mathbf{P}$ is in $\mathcal{O}(D/m)$. In this context, one can hope for algorithms with cost in $\mathcal{O}\tilde{}(m^\omega\lceil D/m\rceil)$.

Let $\mathbf{P}$ be such that $\deg(\det(\mathbf{P})) \leqslant D$ and be $\mathbf{s}$-reduced for $\mathbf{s}$ satisfying $\mathcal{H}_{\mathbf{s},2}$. Since $\mathrm{lm}_{\mathbf{s}}(\mathbf{P})$ is invertible, we can find a permutation π of $\{1, \ldots, m\}$ such that its entries at indices $\{(\pi(j), j), 1 \leqslant j \leqslant m\}$ are nonzero. Up to permuting the rows of $\mathbf{P}$ by π, we can assume that the diagonal of $\mathrm{lm}_{\mathbf{s}}(\mathbf{P})$ is nonzero. Then, denoting by $[d_{ij}]_{i,j}$ the degrees of the entries of $\mathbf{P}$, we have $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}) = (d_{11} + s_1, \ldots, d_{mm} + s_m)$, $\deg(\det(\mathbf{P})) = d_{11} + \cdots + d_{mm}$, and for all i, j, $d_{ij} \leqslant d_{ii} + s_i - s_j \leqslant d_{ii} + \max(\mathbf{s}) - s_j$. While d_{ij} may be $-\infty$ when $i \neq j$, the right-hand side is nonnegative. We finally obtain that

$$\sum_{i,j:\, d_{ij} \neq -\infty} d_{ij} \;\leqslant\; \sum_{i,j} (d_{ii} + \max(\mathbf{s}) - s_j) \;\leqslant\; m \deg(\det(\mathbf{P})) + m|\max(\mathbf{s}) - \mathbf{s}| \;\in\; \mathcal{O}(mD),$$

giving the announced bound on the average degree of $\mathbf{P}$.

For some problems, algorithms have been designed especially to work efficiently under either $\mathcal{H}_{\mathbf{s},1}$ or $\mathcal{H}_{\mathbf{s},2}$ [Sto06, ZL12]. Yet, these assumptions are restrictive, and in fact there

is an important example of a shift which they do not cover (while the ideas in Section 1.2.1 will allow us to deal with it efficiently):

$$\mathbf{h} = (0, D, 2D, \ldots, (m-1)D). \tag{1.4}$$

If a matrix $\mathbf{P}$ is in $\mathbf{h}$-reduced form and such that $\deg(\det(\mathbf{P})) \leqslant D$, then $\mathbf{P}$ is lower triangular; and if $\mathbf{P}$ is furthermore in $\mathbf{h}$-Popov form then it is in Hermite form [BLV06, Lemma 2.6]. The next paragraph proves that this shift is extremal: knowing the bound D, one can restrict to considering shifts that, after being sorted in non-decreasing order, lie between the uniform shift $\mathbf{0}$ and this Hermite shift $\mathbf{h}$, for componentwise comparison.

Reducing the entries of the shift. Let $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$ be nonsingular, let $\mathbf{s} \in \mathbb{Z}^m$, and consider $D \in \mathbb{Z}_{\geqslant 0}$ such that $D > \deg(\det(\mathbf{A}))$. Here, we show how to construct from $\mathbf{s}$ and D a shift $\mathbf{t} \in \mathbb{Z}^m_{\geqslant 0}$ such that

- the $\mathbf{s}$-Popov form $\mathbf{P}$ of $\mathbf{A}$ is also in $\mathbf{t}$-Popov form;
- $\min(\mathbf{t}) = 0$, $\max(\mathbf{t}) \leqslant (m-1)D$, and $|\mathbf{t}| \leqslant m^2 D/2$.

We write $\hat{\mathbf{s}} = (s_{\pi(1)}, \ldots, s_{\pi(m)})$, with π any permutation of $\{1, \ldots, m\}$ such that $\hat{\mathbf{s}}$ is non-decreasing. Then, we define $\hat{\mathbf{t}} = (\hat{t}_1, \ldots, \hat{t}_m)$ by $\hat{t}_1 = 0$ and, for $2 \leqslant i \leqslant m$,

$$\hat{t}_i - \hat{t}_{i-1} = \begin{cases} D & \text{if } \hat{s}_i - \hat{s}_{i-1} \geqslant D, \\ \hat{s}_i - \hat{s}_{i-1} & \text{otherwise.} \end{cases}$$

Let $\mathbf{t} = (\hat{t}_{\pi^{-1}(1)}, \ldots, \hat{t}_{\pi^{-1}(m)})$. Since the diagonal entries of $\mathbf{P}$ have degree at most $\deg(\det(\mathbf{A})) < D$, it is easily verified that $\mathbf{P}$ is in $\mathbf{t}$-ordered weak Popov form. Furthermore, since $\mathbf{P}$ is in $\mathbf{s}$-Popov form, in each column its entries have degree less than the diagonal entry, hence $\mathbf{P}$ is in $\mathbf{t}$-Popov form.

1.3 Gröbner bases of modules over multivariate polynomial rings

In this section, we extend our considerations to modules over a *multivariate* polynomial ring $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X_1, \ldots, X_r]$. Having several variables brings new developments, since it implies that several properties at the core of our discussions in the previous section do not hold anymore. Namely, the fundamental difference is that $\mathbb{K}[\boldsymbol{X}]$ is not a principal ideal domain as soon as $r \geqslant 2$, and as a consequence most $\mathbb{K}[\boldsymbol{X}]$-modules we will consider do not have a basis, consisting of $\mathbb{K}[\boldsymbol{X}]$-linearly independent elements. Still, it is known from Hilbert Basis theorem that submodules of $\mathbb{K}[\boldsymbol{X}]^m$ always have a generating set: this is what we will compute for our problems. New questions arise, starting with that of specifying which useful properties we would like to require on the computed generators.

We will observe that generating sets of a submodule generated by monomials allow us to efficiently compute in the quotient module, and can be directly turned into a generating set which is unique in terms of the given submodule. Then, wanting similar properties

for generating sets of submodules in general, we present the notion of *Gröbner basis*. It fundamentally relies on a chosen *monomial order*, and for a given order a submodule admits a unique *reduced* Gröbner basis. We detail how this generalizes to several variables the shifted Popov bases of univariate modules presented in Section 1.1, where the shift can be seen as the specification of a monomial order.

Finally, we give more details and examples about a specific type of modules that plays an important role in the problems we are interested in. Namely, these are the modules which have *finite dimension* as $\mathbb{K}$-vector spaces. Many computations with this kind of modules can be done by means of linear algebra over $\mathbb{K}$, involving *multiplication matrices*.

1.3.1 Generating sets of ideals and modules

Hereafter, we consider a polynomial ring $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X_1, \ldots, X_r]$ for some integer $r \geqslant 1$. As in Section 1.1, we study $\mathbb{K}[\boldsymbol{X}]$-submodules of $\mathbb{K}[\boldsymbol{X}]^m$. In the case of one variable, we derived many properties of submodules from the fact that a univariate polynomial ring is a principal ideal domain: any ideal of $\mathbb{K}[X]$ is generated by a single polynomial. A notable consequence is that submodules of $\mathbb{K}[X]^m$ are free and of rank at most m, as stated in Lemma 1.1. This does not hold anymore as soon as $r \geqslant 2$. We start with the case $m = 1$, for which the $\mathbb{K}[\boldsymbol{X}]$-submodules of $\mathbb{K}[\boldsymbol{X}]$ are exactly the ideals of $\mathbb{K}[\boldsymbol{X}]$.

Ideals. In $\mathbb{K}[X]$, the question of finding a generating set of an ideal is naturally interpreted as the question of finding its unique monic generator. The situation is more complex in the multivariate case.

Example 1.29. The ideal of $\mathbb{K}[X, Y]$ generated by X and Y, denoted by $\mathcal{I} = \langle X, Y \rangle$, is not principal. Indeed, if there exists a nonzero generator $f \in \mathbb{K}[X, Y]$ such that $\mathcal{I} = \langle f \rangle$, then both X and Y must be multiples of f, which implies that f is a constant; but then we would have $\mathcal{I} = \mathbb{K}[X, Y]$, which is not the case.

Now, assume that there is a generating set $\{f_1, \ldots, f_s\}$ of $\mathcal{I}$ consisting of $s \geqslant 2$ polynomials. Then, f_1 and f_2 are not $\mathbb{K}[\boldsymbol{X}]$-linearly independent, since a nontrivial combination $\alpha f_1 + \beta f_2 = 0$ is given by $\alpha = f_2$ and $\beta = -f_1$. ✍

One may note that the latter paragraph holds more generally for any ideal $\mathcal{I}$ of $\mathbb{K}[\boldsymbol{X}]$: if $\mathcal{I}$ is not principal, it does not have a basis; we recall that we reserved the word *basis* for generating sets that are $\mathbb{K}[\boldsymbol{X}]$-linearly independent. Such relations between generators of $\mathcal{I}$ are called *syzygies*. Here is another example, encountered in interpolation problems such as the one in Example 1.4.

Example 1.30. Let $\{(x_j, y_{j,1}, \ldots, y_{j,r}) \in \mathbb{K}^{r+1}, 1 \leqslant j \leqslant D\}$ be a set of points with $x_1, \ldots, x_D$ pairwise distinct. Then,

$$\mathcal{I} = \{Q \in \mathbb{K}[X, Y_1, \ldots, Y_r] \mid Q(x_j, y_{j,1}, \ldots, y_{j,r}) = 0 \text{ for all } 1 \leqslant j \leqslant D\}$$

is an ideal. Let $M(X) = (X - x_1) \cdots (X - x_D)$, and for each $k \in \{1, \ldots, r\}$ let $L_k(X)$ be in $\mathbb{K}[X]$ such that $L_k(x_j) = y_{j,k}$ for $1 \leqslant j \leqslant D$. Then,

$$\mathcal{I} = \langle M(X), Y_1 - L_1(X), \ldots, Y_r - L_r(X) \rangle.$$

As in the previous example, one can easily prove that $\mathcal{I}$ is not principal, and that it does not have a basis. ✍

Although only principal ideals have a basis, the following result of Hilbert states that all ideals in $\mathbb{K}[\boldsymbol{X}]$ have a finite generating set [Eis95, Theorem 1.2].

Theorem 1.31 (Hilbert Basis Theorem)**.** *Let $\mathcal{I}$ be an ideal of $\mathbb{K}[\boldsymbol{X}]$. Then $\mathcal{I}$ is finitely generated, that is, there exist $f_1, \ldots, f_s \in \mathbb{K}[\boldsymbol{X}]$ such that $\mathcal{I} = \langle f_1, \ldots, f_s \rangle$.*

Note that this is called the *basis* theorem: in this result, as well as in the terminology of Gröbner *bases* introduced below in Section 1.3.3, the word basis commonly refers to a set of generators, without requiring that they be linearly independent over the polynomial ring. To avoid possible ambiguities in the univariate case, which concerns a large part of this document and where the bases we are interested in are linearly independent, in what follows we will not use the word *basis* unless we know that the concerned generating set is linearly independent, except of course in the standard name Gröbner basis.

Modules. Now, we consider the more general case of submodules of $\mathbb{K}[\boldsymbol{X}]^m$ for $m \geqslant 1$. As above for ideals ($m = 1$), such submodules do not have a basis in general. Yet, it follows from the Hilbert Basis Theorem that they are finitely generated [Eis95, Proposition 1.4].

Example 1.32 (Multivariate rational reconstruction [Fit95, Fit97])*.* Let us consider a principal ideal $\mathcal{I} = \langle f \rangle$ of $\mathbb{K}[\boldsymbol{X}]$, and let $g \in \mathbb{K}[\boldsymbol{X}]$. Then,

$$\{(p_1, p_2) \in \mathbb{K}[\boldsymbol{X}]^2 \mid p_1 = p_2 g \bmod f\}$$

is a free submodule of $\mathbb{K}[\boldsymbol{X}]^2$ of rank 2, with basis $\{(g, 1), (f, 0)\}$ [Fit97, Theorem 3.1]. Now, suppose that $\mathcal{I}$ is not principal; then the submodule

$$\mathcal{M} = \{(p_1, p_2) \in \mathbb{K}[\boldsymbol{X}]^2 \mid p_1 + p_2 g \in \mathcal{I}\}$$

is not free. Still, it is finitely generated. Indeed, according to the Hilbert Basis Theorem there exist polynomials $f_1, \ldots, f_s$ such that $\mathcal{I} = \langle f_1, \ldots, f_s \rangle$, and then $\mathcal{M}$ is generated by $\{(g, -1), (f_1, 0), \ldots, (f_s, 0)\}$. ✍

In this context, we will compute generating sets for submodules of $\mathbb{K}[\boldsymbol{X}]^m$. Like in the univariate case, we are interested in such sets that have specific properties. An important family of ideals and modules is formed by those which are generated by monomials.

Monomial modules. Now, we present the definition of monomial submodules and the basic properties of their generating sets. The reader may refer to [Eis95, CLO05, CLO07] for a more detailed introduction, and to [MS05] for an in-depth study.

Let us denote by $\mathbf{c}_1, \ldots, \mathbf{c}_m$ the coordinate vectors

$$\mathbf{c}_j = (0, \ldots, 0, 1, 0, \ldots, 0) \in \mathbb{K}[\boldsymbol{X}]^m \text{ with 1 at index } j,$$

which form a basis of the free $\mathbb{K}[\boldsymbol{X}]$-module $\mathbb{K}[\boldsymbol{X}]^m$. A *monomial* in $\mathbb{K}[\boldsymbol{X}]$ is a product of powers of the variables, $X_1^{i_1} \cdots X_r^{i_r}$ for some $(i_1, \ldots, i_r) \in \mathbb{Z}_{\geqslant 0}^r$. More generally, a monomial

in $\mathbb{K}[\boldsymbol{X}]^m$ is $f\mathbf{c}_j = (0, \ldots, 0, f, 0, \ldots, 0)$ where $1 \leqslant j \leqslant m$ and f is any monomial of $\mathbb{K}[\boldsymbol{X}]$.

Then, a *term* in $\mathbb{K}[\boldsymbol{X}]$ or in $\mathbb{K}[\boldsymbol{X}]^m$ is a monomial multiplied by a scalar from $\mathbb{K}$. In particular, a polynomial $f \in \mathbb{K}[\boldsymbol{X}]^m$ is a finite $\mathbb{K}$-linear combination of monomials in $\mathbb{K}[\boldsymbol{X}]^m$, or in other words a finite sum of terms in $\mathbb{K}[\boldsymbol{X}]^m$; these are called the *terms of* f. Given terms f and g in $\mathbb{K}[\boldsymbol{X}]$, we say that the term $f\mathbf{c}_j$ is *divisible by* the term $g\mathbf{c}_k$ if $j = k$ and f is divisible by g in $\mathbb{K}[\boldsymbol{X}]$.

An important family of ideals of $\mathbb{K}[\boldsymbol{X}]$ is formed by the *monomial ideals*, that is, those generated by monomials of $\mathbb{K}[\boldsymbol{X}]$. Similarly, a *monomial submodule* of $\mathbb{K}[\boldsymbol{X}]^m$ is a $\mathbb{K}[\boldsymbol{X}]$-submodule of $\mathbb{K}[\boldsymbol{X}]^m$ which is generated by monomials of $\mathbb{K}[\boldsymbol{X}]^m$. Note that such a monomial submodule $\mathcal{M}$ can be written $\mathcal{M} = \mathcal{I}_1\mathbf{c}_1 \oplus \cdots \oplus \mathcal{I}_m\mathbf{c}_m$, where $\mathcal{I}_j$ is the monomial ideal of $\mathbb{K}[\boldsymbol{X}]$ generated by the monomials f such that $f\mathbf{c}_j \in \mathcal{M}$.

Generating sets of monomial submodules have particularly nice properties. Let $\mathcal{M} = \langle f_1\mathbf{c}_{j_1}, \ldots, f_s\mathbf{c}_{j_s}\rangle$ be a monomial submodule. Then, we have a straightforward membership characterization: an element $f \in \mathbb{K}[\boldsymbol{X}]^m$ belongs to $\mathcal{M}$ if and only if every term of f is divisible by one of the monomials $f_1\mathbf{c}_{j_1}, \ldots, f_s\mathbf{c}_{j_s}$. As a consequence, a basis of the quotient $\mathbb{K}[\boldsymbol{X}]^m/\mathcal{M}$ as a $\mathbb{K}$-vector space is given by the set of monomials not in $\mathcal{M}$.

Furthermore, one can easily turn the above set of generators of $\mathcal{M}$ into a unique, minimal set of generators, by repeatedly removing from the set any monomial that is divisible by others. The obtained generating set contains only elements that are minimal for the partial order defined by divisibility on the monomials of $\mathbb{K}[\boldsymbol{X}]^m$.

Identifying the exponents of the monomials with tuples in $\mathbb{Z}^r_{\geqslant 0}$ allows us to draw them, and observe the properties stated above on this picture. In Fig. 1.1, we show examples of monomial ideals in two variables; an example in three variables is given in Fig. 1.2.

To summarize, generating sets of monomial submodules allow us to easily test membership, work in the quotient $\mathbb{K}[\boldsymbol{X}]^m/\mathcal{M}$, or find a canonical generating set. For a submodule $\mathcal{M}$ in general, generating sets do not have such properties. Still, there is a specific type of generating set which has them, called a Gröbner basis of $\mathcal{M}$, which besides reduces many questions about $\mathcal{M}$ to questions about a monomial submodule related to $\mathcal{M}$: the initial module of $\mathcal{M}$.

1.3.2 Monomial orders and initial module

As above, let $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X_1, \ldots, X_r]$ and $m \in \mathbb{Z}_{>0}$. Here, we first introduce monomial orders on $\mathbb{K}[\boldsymbol{X}]^m$, which then allow us to associate to any submodule $\mathcal{M}$ of $\mathbb{K}[\boldsymbol{X}]^m$ its initial module. This is a monomial submodule which contains important information about $\mathcal{M}$ and will play a central role in the definition of a Gröbner basis of $\mathcal{M}$ in Section 1.3.3.

Monomial orders. First, we present monomial orders, mainly through several important examples. For more details and insight, we refer the reader to [Eis95, Section 15.3], [KR00, Sections 1.4 and 1.5], [CLO05, Section 5.2], and [CLO07, Section 2.2].

Definition 1.33 (Monomial order)**.** *A* monomial order on $\mathbb{K}[\boldsymbol{X}]^m$ *is a total order* $\prec$ *on the monomials of* $\mathbb{K}[\boldsymbol{X}]^m$ *such that, for any pair of monomials* f, g *of* $\mathbb{K}[\boldsymbol{X}]^m$ *and any*

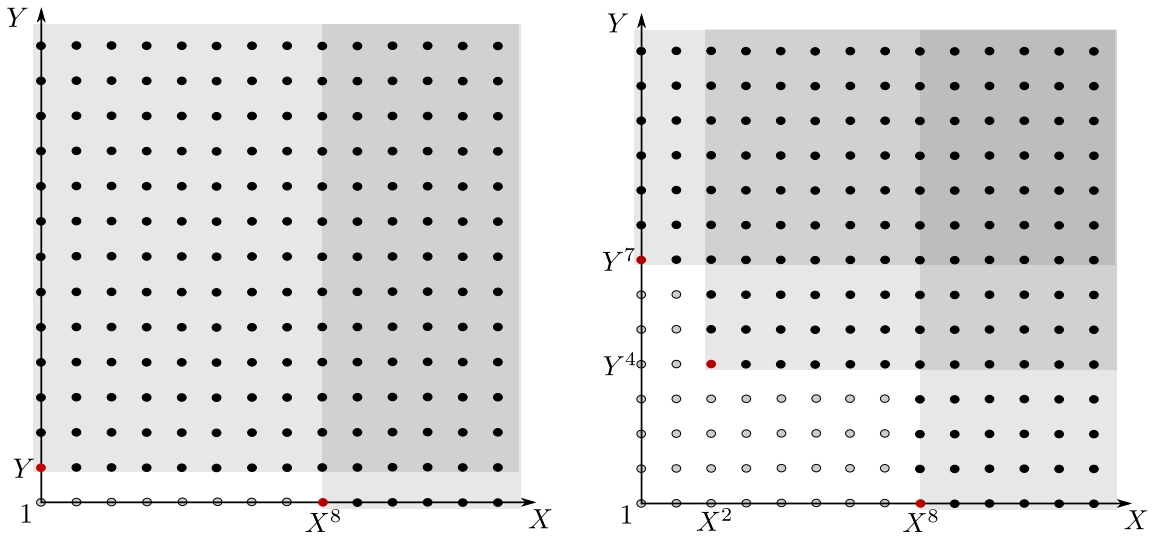

Figure 1.1: The *staircase* of the bivariate monomial ideals $\langle X^8, Y\rangle$ and $\langle X^8, X^2Y^4, Y^7\rangle$. An arbitrary generating set for these monomial ideals must contain the minimal generators (red dots); apart from that, it may contain any finite subset of the other monomials in the ideal, which corresponds to the greyed part.

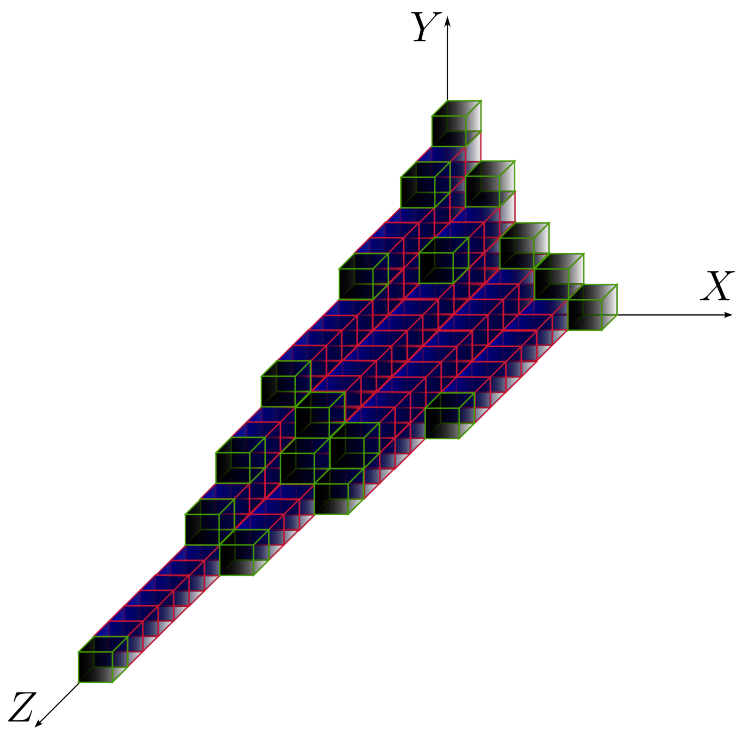

Figure 1.2: The *staircase* of a monomial ideal in three variables. The minimal generating set is indicated by black-green cubes, while the monomial basis of the quotient module is indicated by blue-red cubes.

monomial $h \neq 1$ *of* $\mathbb{K}[\boldsymbol{X}]$,

$$f \prec g \text{ implies } f \prec hf \prec hg.$$

Let us give examples, first considering the case of the multivariate polynomial ring, that is, $m = 1$. Note that in the case of one variable, this definition allows only one monomial order on $\mathbb{K}[X]$, which is the natural order by comparing the exponents: $X^i \prec X^j$ if and only if $i < j$.

Hereafter, we only consider monomial orders such that $X_r \prec \cdots \prec X_1$; this can be ensured without loss of generality up to renaming the variables. One first example is the order used to sort the entries in a dictionary.

Example 1.34. The *lexicographic order* $\prec_{\text{lex}}$ on the monomials of $\mathbb{K}[\boldsymbol{X}]$ is defined as follows, for two monomials $f = X_1^{i_1} \cdots X_r^{i_r}$ and $g = X_1^{j_1} \cdots X_r^{j_r}$:

$$f \prec_{\text{lex}} g \text{ if } i_k < j_k \text{ for the first index } k \text{ with } i_k \neq j_k.$$

For example, $X_1^2 X_2^3 \prec_{\text{lex}} X_1^3 X_2^2 \prec_{\text{lex}} X_1^3 X_2^5 \prec_{\text{lex}} X_1^4$. ✍

For convenience, we will sometimes identify monomials and the corresponding exponent tuple, and consider monomial orders as orders on $\mathbb{Z}_{\geqslant 0}^r$. In the previous example, for $r = 2$, we would write $(2,3) \prec_{\text{lex}} (3,2) \prec_{\text{lex}} (3,5)$. Another monomial order, which has proven to be very important in Gröbner basis computations, is the following.

Example 1.35. The *degree reverse lexicographic order* $\prec_{\text{drl}}$ on $\mathbb{K}[\boldsymbol{X}]$ is defined as follows, for two monomials $f = X_1^{i_1} \cdots X_r^{i_r}$ and $g = X_1^{j_1} \cdots X_r^{j_r}$:

$$\begin{aligned} f \prec_{\text{drl}} g \text{ if } \deg(f) < \deg(g), \text{ or if } \deg(f) = \deg(g) \text{ and} \\ i_k > j_k \text{ for the last index } k \text{ with } i_k \neq j_k, \end{aligned}$$

where $\deg(\cdot)$ stands for the total degree. For example, $X_1^2 X_2^3 \prec_{\text{drl}} X_2^6 \prec_{\text{drl}} X_1^2 X_2^4$. ✍

This is an example of an order which refines the degree. In general, a monomial order $\prec$ on $\mathbb{K}[\boldsymbol{X}]$ is said to *refine the degree* if it first compares the total degree of monomials and then uses another tie-breaking order in case of equality.

Now we give the most often encountered monomial orders for modules. These are two natural extensions of monomial orders on $\mathbb{K}[\boldsymbol{X}]$ to monomial orders on $\mathbb{K}[\boldsymbol{X}]^m$, called *term-over-position* and *position-over-term*, respectively. We also include a shifted variant of term-over-position orders, which can be found for example in [Fit97] and [Mid11, Definition 6.19], and which will allow us in Section 1.3.3 to establish precise links with notions introduced in the univariate case in Section 1.1.

Definition 1.36 (TOP order, POT order)**.** *Let* $\prec$ *be a monomial order on* $\mathbb{K}[\boldsymbol{X}]$*. Consider the module* $\mathbb{K}[\boldsymbol{X}]^m$ *with its canonical basis* $\mathbf{c}_1, \ldots, \mathbf{c}_m$*, and let* $S_1, \ldots, S_m$ *be monomials in* $\mathbb{K}[\boldsymbol{X}]$*. Then,* $\prec$ *induces the following monomial orders on* $\mathbb{K}[\boldsymbol{X}]^m$*. For any pair of monomials* $f\mathbf{c}_j$ *and* $g\mathbf{c}_k$ *of* $\mathbb{K}[\boldsymbol{X}]^m$*,*

- $\prec$-POT order: $f\mathbf{c}_j \prec^{\text{pot}} g\mathbf{c}_k$ *if* $j < k$*, or if* $j = k$ *and* $f \prec g$*;*

- $\prec$-TOP order: $f\mathbf{c}_j \prec^{\text{top}} g\mathbf{c}_k$ *if* $f \prec g$, *or if* $f = g$ *and* $j < k$;
- shifted $\prec$-TOP order: $f\mathbf{c}_j \prec^{S\text{-top}} g\mathbf{c}_k$ *if* $fS_j \prec gS_k$, *or if* $fS_j = gS_k$ *and* $j < k$.

For a more complete treatment of monomial orders for polynomials rings and modules, one may refer to [Rob86, Eis95, FG06]. The shifted orders are sometimes called *weighted* orders [FF92, OF07]. Here, we will prefer *shift* for several reasons: this announces links with the notion of shifted degrees from Section 1.1; weighted orders also commonly refer to assigning weights to the variables in the context of monomial orders on $\mathbb{K}[\boldsymbol{X}]$; we will use the latter notion of weights later in multivariate interpolation problems (Problem 11).

Example 1.37. Here, $r = m = 2$. For any monomial order $\prec$ on $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X_1, X_2]$ and any monomials f and g of $\mathbb{K}[\boldsymbol{X}]$, we have $(f, 0) \prec^{\text{pot}} (0, g)$ and $(f, 0) \prec^{\text{top}} (0, f)$. Furthermore, $(f, 0) \prec^{\text{pot}} (g, 0)$ and $(f, 0) \prec^{\text{top}} (g, 0)$ are both equivalent to $f \prec g$. For $\prec^{\text{top}}_{\text{lex}}$, we have

$$(0, X_1) \prec^{\text{top}}_{\text{lex}} (X_1^2X_2^3, 0) \prec^{\text{top}}_{\text{lex}} (0, X_1^3X_2^2) \prec^{\text{top}}_{\text{lex}} (X_1^3X_2^5, 0) \prec^{\text{top}}_{\text{lex}} (0, X_1^3X_2^5),$$

while for $\prec^{\text{pot}}_{\text{lex}}$ we have

$$(X_1^2X_2^3, 0) \prec^{\text{pot}}_{\text{lex}} (X_1^3X_2^5, 0) \prec^{\text{pot}}_{\text{lex}} (0, X_1) \prec^{\text{pot}}_{\text{lex}} (0, X_1^3X_2^2) \prec^{\text{pot}}_{\text{lex}} (0, X_1^3X_2^5). \qquad ✍$$

Example 1.38. Here, we consider the univariate module $\mathbb{K}[X]^m$, that is, $r = 1$ and $m \geqslant 1$. As mentioned above, in the case of univariate polynomials, the only monomial order over $\mathbb{K}[X]$ is given by $X^a < X^b \Leftrightarrow a < b$. Furthermore, the shifting monomials are $S_j = X^{s_j}$ for all j, defining a shift $\mathbf{s} = (s_1, \ldots, s_m) \in \mathbb{Z}^m_{\geqslant 0}$. As a consequence, the monomial orders defined above specialize into the following:

- *$<$-POT order:* $X^a\mathbf{c}_j <^{\text{pot}} X^b\mathbf{c}_k$ if and only if $(j, a) \prec_{\text{lex}} (k, b)$;
- *$<$-TOP order:* $X^a\mathbf{c}_j <^{\text{top}} X^b\mathbf{c}_k$ if and only if $(a, j) \prec_{\text{lex}} (b, k)$;
- *shifted $<$-TOP order:* $X^a\mathbf{c}_j <^{\mathbf{s}\text{-top}} X^b\mathbf{c}_k$ if and only if $(a + s_j, j) \prec_{\text{lex}} (b + s_k, k)$.

The reader may note that this is reminiscent of the notions of row degree and pivots from Section 1.1. We will clearly state this link in Example 1.40 and Section 1.3.4. ✍

Initial module. Now that we have defined total orders on the monomials of $\mathbb{K}[\boldsymbol{X}]^m$, we may define the notion of initial term of a polynomial with respect to this order, which generalizes to several variables $r \geqslant 1$ and to modules $m \geqslant 1$ the notion of leading term for univariate polynomials.

For a given monomial order $\prec$ on $\mathbb{K}[\boldsymbol{X}]^m$ and an element $f \in \mathbb{K}[\boldsymbol{X}]^m$, the *$\prec$-initial term of f*, denoted by $\text{in}_\prec(f)$, is the term of f whose monomial is the greatest with respect to $\prec$. We extend this to any collection $\mathcal{F} \subseteq \mathbb{K}[\boldsymbol{X}]^m$ of polynomials: the $\prec$-initial terms of $\mathcal{F}$, denoted by $\text{in}_\prec(\mathcal{F})$, is the set of initial terms $\{\text{in}_\prec(f), f \in \mathcal{F}\}$ of the elements of $\mathcal{F}$. For a module $\mathcal{M}$, the *$\prec$-initial module of $\mathcal{M}$*, denoted by $\text{in}_\prec(\mathcal{M})$, is the monomial submodule $\{\text{in}_\prec(f), f \in \mathcal{M}\}$ of $\mathbb{K}[\boldsymbol{X}]^m$.

For example, the $\prec_{\text{lex}}$-initial term of $f = 3X_1^3X_2^5 + 5X_1^3X_2^2 + 2X_1^2X_2^3 \in \mathbb{K}[X_1, X_2]$ is $\text{in}_{\prec_{\text{lex}}}(f) = 3X_1^3X_2^5$.

Example 1.39 (Multivariate rational reconstruction). From Example 1.32, for a given ideal $\mathcal{I} = \langle f_1, \ldots, f_s \rangle$ of $\mathbb{K}[\boldsymbol{X}]$ and some element $g \in \mathbb{K}[\boldsymbol{X}]$,

$$\mathcal{M} = \{(p_1, p_2) \in \mathbb{K}[\boldsymbol{X}]^2 \mid p_1 + p_2 g \in \mathcal{I}\}$$

is a submodule of $\mathbb{K}[\boldsymbol{X}]^2$ generated by $\mathcal{F} = \{(-g, 1), (f_1, 0), \ldots, (f_s, 0)\}$. Let $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]$. Then, assuming g is nonconstant, for the $\prec$-TOP order we have

$$\mathrm{in}_{\prec^{\mathrm{top}}}(\mathcal{F}) = \{(\mathrm{in}_{\prec}(g), 0), (\mathrm{in}_{\prec}(f_1), 0), \ldots, (\mathrm{in}_{\prec}(f_s), 0)\}.$$

In particular $\langle \mathrm{in}_{\prec^{\mathrm{top}}}(\mathcal{F}) \rangle$ is a strict subset of $\mathrm{in}_{\prec^{\mathrm{top}}}(\mathcal{M})$, since we have for example $(0, f_1) = f_1(g, 1) - g(f_1, 0) \in \mathcal{M}$ while $\mathrm{in}_{\prec^{\mathrm{top}}}(0, f_1) = (0, \mathrm{in}_{\prec}(f_1)) \notin \langle \mathrm{in}_{\prec^{\mathrm{top}}}(\mathcal{F}) \rangle$. On the other hand, for the $\prec$-POT order we have

$$\mathrm{in}_{\prec^{\mathrm{pot}}}(\mathcal{F}) = \{(0, -1), (\mathrm{in}_{\prec}(f_1), 0), \ldots, (\mathrm{in}_{\prec}(f_s), 0)\},$$

and one can check that in this case $\langle \mathrm{in}_{\prec^{\mathrm{pot}}}(\mathcal{F}) \rangle = \mathrm{in}_{\prec}(\mathcal{I}) \times \mathbb{K}[\boldsymbol{X}] = \mathrm{in}_{\prec^{\mathrm{pot}}}(\mathcal{M})$. ✍

The latter situation, when we have a generating set of $\mathcal{M}$ whose initial terms generate the initial module of $\mathcal{M}$, indicates that this generating set has good properties; this will be developed in Section 1.3.3. In the univariate case, we can relate the initial term with the notion of pivot introduced in Definition 1.13.

Example 1.40. Consider the $\mathbb{K}[X]$-module $\mathbb{K}[X]^{1\times m}$, and let $<$ denote the monomial order on $\mathbb{K}[X]$. For a shift $\mathbf{s} \in \mathbb{Z}^m$ and a nonzero polynomial vector $\mathbf{p} = [p_j]_j \in \mathbb{K}[X]^{1\times m}$, let $(0, \ldots, 0, \alpha X^d, 0, \ldots, 0)$ denote the $<^{\mathbf{s}\text{-top}}$-initial term of $\mathbf{p}$, with αX^d at index π. Then the $\mathbf{s}$-pivot entry of $\mathbf{p}$ is p_π, the $\mathbf{s}$-pivot degree of $\mathbf{p}$ is $d = \deg(p_\pi)$, and the leading term of p_π is αX^d.

More generally, let $\mathcal{M}$ be a submodule of $\mathbb{K}[X]^{1\times m}$ of rank m, let $\mathbf{s} \in \mathbb{Z}^m$, and let $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m)$ be the $\mathbf{s}$-minimal degree of $\mathcal{M}$. Then the $<^{\mathbf{s}\text{-top}}$-initial module of $\mathcal{M}$ is

$$\mathrm{in}_{<^{\mathbf{s}\text{-top}}}(\mathcal{M}) = X^{\delta_1}\mathbb{K}[X] \times \cdots \times X^{\delta_m}\mathbb{K}[X].$$

We recall that, as noted in Section 1.1.3, the quotient $\mathbb{K}[X]^{1\times m}/\mathcal{M}$ is isomorphic to the product of residue class rings $\mathbb{K}[X]/\langle X^{\delta_1} \rangle \times \cdots \times \mathbb{K}[X]/\langle X^{\delta_m} \rangle$, and it admits the vector space basis $\cup_{1 \leqslant j \leqslant m} \{X^d \mathbf{c}_j, 0 \leqslant d < \delta_j\}$ where $\mathbf{c}_j \in \mathbb{K}[X]^{1\times m}$ is the coordinate vector with 1 at index j. ✍

The last property in this example actually holds more generally, as stated in the next theorem. This implies in particular that the initial module gives important information on the quotient $\mathbb{K}[\boldsymbol{X}]^m/\mathcal{M}$, notably for performing computations therein.

Theorem 1.41 (Macaulay). *Let $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X_1, \ldots, X_r]$, let $m \in \mathbb{Z}_{>0}$, and let $\mathcal{M}$ be a $\mathbb{K}[\boldsymbol{X}]$-submodule of $\mathbb{K}[\boldsymbol{X}]^m$. For any monomial order $\prec$ on $\mathbb{K}[\boldsymbol{X}]^m$, the set $\mathcal{E}$ of all monomials not in $\mathrm{in}_{\prec}(\mathcal{M})$ forms a basis of $\mathbb{K}[\boldsymbol{X}]^m/\mathcal{M}$ as a $\mathbb{K}$-vector space.*

A proof can be found in [Eis95, Section 15.2]. Note that here and hereafter, we identify the indeterminate X_i in $\mathbb{K}[\boldsymbol{X}]$ and its image in the quotient $\mathbb{K}[\boldsymbol{X}]^m/\mathcal{M}$. Then, if the module $\mathbb{K}[\boldsymbol{X}]^m/\mathcal{M}$ is finite-dimensional as a $\mathbb{K}$-vector space, it admits a finite basis $\mathcal{E} = \{\varepsilon_1, \ldots, \varepsilon_D\}$ formed by the monomials not in $\mathrm{in}_{\prec}(\mathcal{M})$, which is commonly called *the $\prec$-monomial basis of* $\mathbb{K}[\boldsymbol{X}]^m/\mathcal{M}$. Note that it is sometimes required that this basis be ordered such that $1 = \varepsilon_1 \prec \cdots \prec \varepsilon_D$.

Example 1.42. Following the notation of Example 1.30, let us show in detail that

$$\begin{aligned}\mathcal{I} &= \{Q \in \mathbb{K}[X, Y_1, \dots, Y_r] \mid Q(x_j, y_{j,1}, \dots, y_{j,r}) = 0 \text{ for all } 1 \leqslant j \leqslant D\} \\ &= \langle M(X), Y_1 - L_1(X), \dots, Y_r - L_r(X)\rangle\end{aligned}$$

is such that the quotient ring $\mathcal{M} = \mathbb{K}[X, Y_1, \dots, Y_r]/\mathcal{I}$ has finite dimension D as a $\mathbb{K}$-vector space, and more precisely, that it admits $\{1, X, \dots, X^{D-1}\}$ as a vector space basis. We will show later in Example 1.45 that, for a certain monomial order $\prec$, the initial ideal of $\mathcal{I}$ is $\mathrm{in}_\prec(\mathcal{I}) = \langle X^D, Y_1, \dots, Y_r\rangle$.

First, the monomials $1, X, \dots, X^{D-1}$ are $\mathbb{K}$-linearly independent in $\mathcal{M}$. Indeed, consider a linear combination $p(X) = p_0 1 + p_1 X + \cdots + p_{D-1} X^{D-1}$ for some $p_0, \dots, p_{D-1} \in \mathbb{K}$. If this combination is zero in the quotient, this means that $p(X) \in \mathcal{I}$, and therefore $p(x_1) = \cdots = p(x_D) = 0$. Thus $p(X)$ is a univariate polynomial of degree less than D which has at least D pairwise distinct roots: it must be zero. Hence $p_0 = \cdots = p_{D-1} = 0$.

Then, the conclusion follows from the remark that for any $f \in \mathbb{K}[X, Y_1, \dots, Y_r]$, we have $f(X, Y_1, \dots, Y_r) = f(X, L_1, \dots, L_r) + g$ for some $g \in \mathcal{I}$. Considering the univariate polynomial $p(X)$ which is the remainder in the (univariate) division of $f(X, L_1, \dots, L_r)$ by $M(X)$, we obtain that the image of f in the quotient $\mathcal{M}$ is the same as that of p, which is a $\mathbb{K}$-linear combination of the monomials $1, X, \dots, X^{D-1}$. ✍

1.3.3 Gröbner bases

As above, let $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X_1, \dots, X_r]$, $m \in \mathbb{Z}_{>0}$, and let $\mathcal{M}$ be a submodule of $\mathbb{K}[\boldsymbol{X}]^m$. From Theorem 1.41, we obtain the following property regarding division modulo $\mathcal{M}$: any polynomial $f \in \mathbb{K}[\boldsymbol{X}]^m$ can be uniquely written $f = g + h$, where $g \in \mathcal{M}$ and h is a $\mathbb{K}$-linear combination of the monomials not in $\mathrm{in}_\prec(\mathcal{M})$. This remainder h is called the *$\prec$-normal form of f (with respect to $\mathcal{M}$)*, denoted by $\mathrm{nf}_\prec(f)$. Then, a natural question is how to find $\mathrm{nf}_\prec(f)$ from f, assuming that $\mathcal{M}$ is given by a generating set $\{f_1, \dots, f_s\}$.

There is an algorithm for performing multivariate division with remainder, which generalizes the Euclidean algorithm for the univariate case, and relies on a monomial order $\prec$ to decide at each step which monomial should be considered. Given $f \in \mathbb{K}[\boldsymbol{X}]^m$, this algorithm produces an expression $f = q_1 f_1 + \cdots + q_s f_s + h$, where none of the terms of h is in $\langle \mathrm{in}_\prec(f_1), \dots, \mathrm{in}_\prec(f_s)\rangle$; for more details, one may refer to [Eis95, Section 15.3] or to [CLO07, Chapter 2, §3].

One problem is that this does not always yield the unique remainder $\mathrm{nf}_\prec(f)$ as above; in other words, h is not unique and we may have $h \neq \mathrm{nf}_\prec(f)$. Indeed, as observed in Example 1.39, $\langle \mathrm{in}_\prec(f_1), \dots, \mathrm{in}_\prec(f_s)\rangle$ may be strictly contained in $\mathrm{in}_\prec(\mathcal{M})$. This leads us to the following definition of a specific type of generating sets of $\mathcal{M}$, called Gröbner bases.

Definition 1.43 (Gröbner basis)**.** *Let $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^m$ and let $\mathcal{M}$ be a $\mathbb{K}[\boldsymbol{X}]$-submodule of $\mathbb{K}[\boldsymbol{X}]^m$. A generating set $\{f_1, \dots, f_s\}$ of $\mathcal{M}$ is said to be a* $\prec$-Gröbner basis *of $\mathcal{M}$ if the $\prec$-initial module of $\mathcal{M}$ is generated by $\{\mathrm{in}_\prec(f_1), \dots, \mathrm{in}_\prec(f_s)\}$, that is, $\mathrm{in}_\prec(\mathcal{M}) = \langle \mathrm{in}_\prec(f_1), \dots, \mathrm{in}_\prec(f_s)\rangle$.*

We will drop from our notation the monomial order $\prec$ when there is no ambiguity. The next result sometimes helps to prove that two sets of polynomials generate the same mod-

ule, by reducing the question to the comparison of their initial modules. This highlights the fact that the initial module contains useful information about the module itself.

Lemma 1.44. *If $\mathcal{N} \subseteq \mathcal{M} \subseteq \mathbb{K}[\boldsymbol{X}]^m$ are two submodules such that $\mathrm{in}_\prec(\mathcal{N}) = \mathrm{in}_\prec(\mathcal{M})$, then $\mathcal{N} = \mathcal{M}$.*

The proof can be found in [Eis95, Lemma 15.5]. An important consequence is that, for any submodule $\mathcal{M}$ of $\mathbb{K}[\boldsymbol{X}]^m$ and any monomial order $\prec$ on $\mathbb{K}[\boldsymbol{X}]^m$, there exists a $\prec$-Gröbner basis of $\mathcal{M}$. Indeed, the initial module of $\mathcal{M}$ is finitely generated: $\mathrm{in}_\prec(\mathcal{M}) = \langle g_1, \dots, g_s \rangle$ for some monomials $g_1, \dots, g_s \in \mathrm{in}_\prec(\mathcal{M})$. These monomials are initial terms of elements $f_1, \dots, f_s \in \mathcal{M}$, and therefore $\mathrm{in}_\prec(\mathcal{M}) = \langle g_1, \dots, g_s \rangle \subseteq \mathrm{in}_\prec(\langle f_1, \dots, f_s \rangle) \subseteq \mathrm{in}_\prec(\mathcal{M})$. Thus, by the above property we have $\mathcal{M} = \langle f_1, \dots, f_s \rangle$.

This lemma also gives us a Gröbner basis test: a set of elements of $\mathcal{M}$ is a $\prec$-Gröbner basis of $\mathcal{M}$ if and only if their $\prec$-initial terms generate $\mathrm{in}_\prec(\mathcal{M})$. Another Gröbner basis test is Buchberger's criterion [Eis95, Theorem 15.8]. The latter can be turned into an algorithm to compute a $\prec$-Gröbner basis of $\mathcal{M}$, for any monomial order $\prec$ and any submodule $\mathcal{M}$ of $\mathbb{K}[\boldsymbol{X}]^m$ [Buc76] (see also [Eis95, Algorithm 15.9]).

Similarly to what we presented above for monomial modules, one can easily transform a $\prec$-Gröbner basis of $\mathcal{M}$ into a minimal one, by removing elements whose initial term is divisible by the initial term of another element. More precisely, $\{f_1, \dots, f_s\}$ is a *minimal* $\prec$-Gröbner basis of $\mathcal{M}$ if $\mathrm{in}_\prec(f_i)$ is monic for all i (the coefficient from $\mathbb{K}$ is 1), and for all $i \neq j$, $\mathrm{in}_\prec(f_i)$ does not divide $\mathrm{in}_\prec(f_j)$ [AL94, Definition 1.8.1]. Although this does not yield uniqueness of the set of generators, all minimal $\prec$-Gröbner bases of $\mathcal{M}$ have the same number of generators and the same initial terms, the latter forming the minimal generating set of $\mathrm{in}_\prec(\mathcal{M})$.

There is a specific minimal $\prec$-Gröbner basis of $\mathcal{M}$, called the *reduced* $\prec$-Gröbner basis of $\mathcal{M}$, which is canonical: it is uniquely defined in terms of the module $\mathcal{M}$ and the monomial order $\prec$. Namely, this is the Gröbner basis $\{f_1, \dots, f_s\}$ of $\mathcal{M}$ which satisfies

- for $1 \leqslant i \leqslant s$, $\mathrm{in}_\prec(f_i)$ is monic;
- for $1 \leqslant i \leqslant s$, $\mathrm{in}_\prec(f_i)$ does not divide any term of f_j for $j \neq i$.

These properties directly give an algorithm to transform any $\prec$-Gröbner basis of $\mathcal{M}$ into the reduced $\prec$-Gröbner basis of $\mathcal{M}$.

Example 1.45. As in Examples 1.30 and 1.42, consider the ideal

$$\mathcal{I} = \{Q \in \mathbb{K}[X, Y_1, \dots, Y_r] \mid Q(x_j, y_{j,1}, \dots, y_{j,r}) = 0 \text{ for all } 1 \leqslant j \leqslant D\}$$

of $\mathbb{K}[X, Y_1, \dots, Y_r]$, and let $M(X) = (X - x_1) \cdots (X - x_D)$, as well as $L_k(X) \in \mathbb{K}[X]$ be such that $L_k(x_j) = y_{j,k}$ for $1 \leqslant j \leqslant D$ and $k \in \{1, \dots, r\}$. Furthermore, let $\prec_{\mathrm{lex}}$ stand for the lexicographic order on $\mathbb{K}[X, Y_1, \dots, Y_r]$, where the variables are ordered arbitrarily with $X \prec_{\mathrm{lex}} Y_j$ for all j. Then,

$$\{M(X), Y_1 - L_1(X), \dots, Y_r - L_r(X)\}$$

is the reduced $\prec_{\mathrm{lex}}$-Gröbner basis of $\mathcal{I}$. ✍

We have the following characterization, which gives in particular a property similar to the predictable-degree property of Theorem 1.11.

Lemma 1.46. *Let $\mathcal{M}$ be a submodule of $\mathbb{K}[\boldsymbol{X}]^m$, $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^m$, and $\mathcal{G} = \{f_1, \dots, f_s\} \subseteq \mathcal{M}$. Then, $\mathcal{G}$ is a $\prec$-Gröbner basis of $\mathcal{M}$ if and only if for all $f \in \mathcal{M}$, there exist $g_1, \dots, g_s \in \mathbb{K}[\boldsymbol{X}]$ such that*

$$f = g_1 f_1 + \cdots + g_s f_s \quad \textit{and} \quad \mathrm{in}_\prec(f) = \max_\prec\{\mathrm{in}_\prec(g_i f_i), 1 \leqslant i \leqslant s\}.$$

For a proof of this result and other characterizations, see for example [AL94, Theorem 3.5.14]. As mentioned above, one of the motivations for introducing Gröbner bases is their usefulness regarding computations in the quotient $\mathbb{K}[\boldsymbol{X}]^m/\mathcal{M}$, or the computation of normal forms $\mathrm{nf}_\prec(f)$ of elements $f \in \mathbb{K}[\boldsymbol{X}]^m$ with respect to $\mathcal{M}$. We summarize properties about division with remainder here.

Lemma 1.47. *Let $\mathcal{M}$ be a submodule of $\mathbb{K}[\boldsymbol{X}]^m$, $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^m$, and $\mathcal{G} = \{f_1, \dots, f_s\} \subseteq \mathcal{M}$ be a $\prec$-Gröbner basis of $\mathcal{M}$. For any $f \in \mathbb{K}[\boldsymbol{X}]^m$, the algorithm of multivariate division with remainder computes $g_1, \dots, g_s \in \mathbb{K}[\boldsymbol{X}]$ and $h \in \mathbb{K}[\boldsymbol{X}]^m$ such that $f = g_1 f_1 + \cdots + g_s f_s + h$, with $h = \mathrm{nf}_\prec(f)$ having no monomial in $\mathrm{in}_\prec(\mathcal{M})$.*

A detailed presentation of this algorithm can be found in [CLO07, Chapter 2, §3]. One may notice the similarity of this result with Lemma 1.24 when working over $\mathbb{K}[X]$, keeping in mind the description of $\mathrm{in}_\prec(\mathcal{M})$ given in Example 1.40. We will now formalize the fact that, in the univariate case, reduced Gröbner bases of submodules of $\mathbb{K}[X]^m$ correspond to its shifted Popov bases.

1.3.4 Link with shifted Popov bases

Here, we focus on the univariate case. In the examples above, we have hinted at strong links between the notions of shift and pivots, and the notions of monomial order and initial terms. Since the former are used to define shifted Popov forms, and the latter are used to define reduced Gröbner bases, one may wonder to which extent these canonical generating sets differ in the case of univariate modules, if they do at all.

In a different context, this connection has been studied in [Mid11, Chapter 6] concerning matrices over Ore polynomial rings. More precisely, [Mid11, Theorem 6.29] indicates that shifted Popov forms coincide with reduced Gröbner bases for shifted TOP orders. This also holds concerning matrices over $\mathbb{K}[X]$; we state this precisely in Lemma 1.49. In particular, we have that $\mathbf{0}$-Popov forms and Hermite forms coincide with reduced Gröbner bases for the TOP order and the POT order, respectively.

We remark that the connection between specific reduced forms of polynomials matrices and Gröbner bases has often been used in the context of coding theory [Fit95, Ale05, LO06, LO08, Tri10], implicitly or explicitly, and was also studied in [Nie13, Section 2.1].

The results in this section can be summarized as follows: *for a given submodule of $\mathbb{K}[X]^m$, shifted Popov bases for arbitrary shifts cover the whole generality of reduced Gröbner bases with respect to arbitrary monomial orders.*

Let $<$ denote the unique monomial order on $\mathbb{K}[X]$, given by $X^a < X^b \Leftrightarrow a < b$. The TOP and POT orders in Definition 1.36 have the property that $\mathbf{c}_1 <^{\mathrm{pot}} \mathbf{c}_2 <^{\mathrm{pot}} \cdots <^{\mathrm{pot}} \mathbf{c}_m$

and $\mathbf{c}_1 <^{\text{top}} \mathbf{c}_2 <^{\text{top}} \cdots <^{\text{top}} \mathbf{c}_m$. An important remark for our discussion here is that, from the definition of a shifted $\mathbf{s}$-TOP order for a given $\mathbf{s} \in \mathbb{Z}^m_{\geqslant 0}$, we have

$$X^{\mu-s_1}\mathbf{c}_1 <^{\mathbf{s}\text{-top}} X^{\mu-s_2}\mathbf{c}_2 <^{\mathbf{s}\text{-top}} \cdots <^{\mathbf{s}\text{-top}} X^{\mu-s_m}\mathbf{c}_m, \tag{1.5}$$

where $\mu = \max(\mathbf{s})$. Thus, one could slightly generalize the definition of the $\mathbf{s}$-TOP order, by using a permutation π of $\{1, \ldots, m\}$ in order to specify how the monomials of the form $X^{\mu-s_i}\mathbf{c}_i$ compare to each other. The result below states that, up to such a permutation, any monomial order on $\mathbb{K}[X]^m$ coincide with a shifted TOP order on the monomials of degree less than a prescribed bound.

Lemma 1.48. *Let $\prec$ be a monomial order on $\mathbb{K}[X]^m$, let $D \in \mathbb{Z}_{>0}$ be some degree bound, and let $\mathbf{c}_1, \ldots, \mathbf{c}_m$ be the canonical basis of $\mathbb{K}[X]^m$. Defining the shift $\mathbf{s} \in \mathbb{Z}^m_{\geqslant 0}$ as detailed below, and denoting by π the permutation of $\{1, \ldots, m\}$ such that $(X^{\max(\mathbf{s})-s_{\pi(i)}}\mathbf{c}_{\pi(i)})_{1\leqslant i\leqslant m}$ is increasing for $\prec$, then we have*

$$X^a\mathbf{c}_i \prec X^b\mathbf{c}_j \;\Leftrightarrow\; a + s_i < b + s_j \text{ or } (a + s_i = b + s_j \text{ and } \pi^{-1}(i) < \pi^{-1}(j))$$

for any pair of monomials $X^a\mathbf{c}_i$ and $X^b\mathbf{c}_j$ of $\mathbb{K}[X]^m$ such that a and b are less than D.

To define $\mathbf{s}$, let ℓ and $\nu_1, \ldots, \nu_\ell$ be the unique positive integers such that $\nu_1 + \cdots + \nu_\ell = m$ and such that one can write $\{1, \ldots, m\} = \{i_{u,v}, 1 \leqslant v \leqslant \nu_u, 1 \leqslant u \leqslant \ell\}$ with

- $\mathbf{c}_{i_{1,1}} \prec \cdots \prec \mathbf{c}_{i_{1,\nu_1}} \prec \cdots \prec \mathbf{c}_{i_{\ell,1}} \prec \cdots \prec \mathbf{c}_{i_{\ell,\nu_\ell}}$,
- $X^D\mathbf{c}_{i_{u,\nu_u}} \prec \mathbf{c}_{i_{u+1,1}}$ *for* $1 \leqslant u \leqslant \ell - 1$,
- $\mathbf{c}_{i_{u,v+1}} \prec X^D\mathbf{c}_{i_{u,v}}$ *for* $1 \leqslant v \leqslant \nu_u - 1$ *and* $1 \leqslant u \leqslant \ell$.

Then, $\mathbf{s} = (s_1, \ldots, s_m) \in \mathbb{Z}^m_{\geqslant 0}$ is defined by

- $s_{i_{1,1}} = 0$ *and* $s_{i_{u+1,1}} = s_{i_{u,\nu_u}} + D$ *for* $1 \leqslant u \leqslant \ell - 1$,
- $s_{i_{u,v}} = s_{i_{u,1}} + \max\{e \mid X^e\mathbf{c}_{i_{u,v}} \prec \mathbf{c}_{i_{u,1}}\}$ *for* $2 \leqslant v \leqslant \nu_u$ *and* $1 \leqslant u \leqslant \ell$.

Proof. Let $X^a\mathbf{c}_{i_{u,v}}$ and $X^b\mathbf{c}_{i_{u',v'}}$ be monomials such that $a, b < D$. We want to prove that $X^a\mathbf{c}_{i_{u,v}} \prec X^b\mathbf{c}_{i_{u',v'}}$ is equivalent to

$$a + s_{i_{u,v}} < b + s_{i_{u',v'}} \text{ or } (a + s_{i_{u,v}} = b + s_{i_{u',v'}} \text{ and } \pi^{-1}(i_{u,v}) < \pi^{-1}(i_{u',v'})).$$

First, if $i_{u,v} = i_{u',v'}$ then the equivalence follows from $X^a\mathbf{c}_{i_{u,v}} \prec X^b\mathbf{c}_{i_{u,v}} \Leftrightarrow a < b$.

Now, if $u < u'$, on the first hand this implies $s_{i_{u,v}} + D \leqslant s_{i_{u',v'}}$, so that $a + s_{i_{u,v}} < b + s_{i_{u',v'}}$ since $a - b < D$; on the other hand, we have $X^{D+a-b}\mathbf{c}_{i_{u,v}} \prec X^{2D}\mathbf{c}_{i_{u,\nu_u}} \prec X^D\mathbf{c}_{i_{u+1,1}} \prec X^D\mathbf{c}_{i_{u',v'}}$, hence $X^a\mathbf{c}_{i_{u,v}} \prec X^b\mathbf{c}_{i_{u',v'}}$. Similarly, if $u' < u$ then $b + s_{i_{u',v'}} < a + s_{i_{u,v}}$ and $X^b\mathbf{c}_{i_{u',v'}} \prec X^a\mathbf{c}_{i_{u,v}}$.

Finally, assume that $u = u'$. First, if $X^a\mathbf{c}_{i_{u,v}} \prec X^b\mathbf{c}_{i_{u,v'}}$ then by definition of $\mathbf{s}$ we have $X^{a+s_{i_{u,v}}-s_{i_{u,1}}}\mathbf{c}_{i_{u,1}} \preceq X^a\mathbf{c}_{i_{u,v}} \prec X^b\mathbf{c}_{i_{u,v'}} \prec X^{b+1+s_{i_{u,v'}}-s_{i_{u,1}}}\mathbf{c}_{i_{u,1}}$, hence $a + s_{i_{u,v}} \leqslant b + s_{i_{u,v'}}$. On the other hand, $a + s_{i_{u,v}} < b + s_{i_{u,v'}}$ implies that $X^a\mathbf{c}_{i_{u,v}} \prec X^{a+1+s_{i_{u,v}}-s_{i_{u,1}}}\mathbf{c}_{i_{u,1}} \preceq$

$X^{b+s_{i_{u,v'}}-s_{i_{u,1}}}\mathbf{c}_{i_{u,1}} \preceq X^b\mathbf{c}_{i_{u,v'}}$. Thus, it remains to show that if $a + s_{i_{u,v}} = b + s_{i_{u,v'}}$, then $X^a\mathbf{c}_{i_{u,v}} \prec X^b\mathbf{c}_{i_{u,v'}} \Leftrightarrow \pi^{-1}(i_{u,v}) < \pi^{-1}(i_{u,v'})$. By definition of π, $\pi^{-1}(i_{u,v}) < \pi^{-1}(i_{u,v'}) \Leftrightarrow X^{\mu-s_{i_{u,v}}}\mathbf{c}_{i_{u,v}} \prec X^{\mu-s_{i_{u,v'}}}\mathbf{c}_{i_{u,v'}}$, where $\mu = \max(\mathbf{s})$; then, multiplying by $X^{a+s_{i_{u,v}}} = X^{b+s_{i_{u,v'}}}$ yields $X^{\mu-s_{i_{u,v}}}\mathbf{c}_{i_{u,v}} \prec X^{\mu-s_{i_{u,v'}}}\mathbf{c}_{i_{u,v'}} \Leftrightarrow X^{\mu+a}\mathbf{c}_{i_{u,v}} \prec X^{\mu+b}\mathbf{c}_{i_{u,v'}}$, hence the equivalence. ☙

Thus, when we restrict to comparing monomials of degree less than D, apart from having fixed π to be the identity permutation, shifted TOP orders in Definition 1.36 represent all possible monomial orders. Similarly, we fixed a permutation to be the identity when we defined the notion of $\mathbf{s}$-pivot of a polynomial vector in Definition 1.13. Indeed, we defined the pivot as the largest index in $\{1, \ldots, m\}$ satisfying some property. This corresponds to ensuring the ordering in Eq. (1.5), via the link between pivots and $<^{\mathbf{s}\text{-top}}$-initial terms detailed in Example 1.40.

The straightforward generalization of shifted TOP orders involving the permutation π then corresponds to the similar generalization of $\mathbf{s}$-pivots, involving a permutation of $\{1, \ldots, m\}$ which specifies which is the $\mathbf{s}$-pivot among several monomials having the same $\mathbf{s}$-degree, instead of requiring that this always be the one with largest index.

This leads us to the next result, which gives an explicit correspondence between Gröbner bases of submodules of $\mathbb{K}[X]^m$ and their bases in specific reduced forms that we defined in Section 1.1. In this lemma, $\mathbf{X}^{\mu-\mathbf{s}} = \operatorname{diag}(X^{\mu-s_1}, \ldots, X^{\mu-s_m})$ is the shift matrix from Section 1.1.2.

Lemma 1.49. *Let $\mathcal{M}$ be a $\mathbb{K}[X]$-submodule of $\mathbb{K}[X]^{1\times m}$ of rank ρ, and let $\prec$ be a monomial order on $\mathbb{K}[X]^{1\times m}$, and let $D \in \mathbb{Z}_{>0}$ be some degree bound. Let $\mathbf{s} \in \mathbb{Z}^m_{\geqslant 0}$ and π be defined from $\prec$ and D as in Lemma 1.48; we see π as the $m \times m$ permutation matrix such that the rows of $\mathbf{X}^{\mu-\mathbf{s}}\pi$ are $\prec$-increasing. Let $\{\mathbf{p}_1, \ldots, \mathbf{p}_\rho\} \subset \mathbb{K}[X]^{1\times m}$ and let $\mathbf{P} \in \mathbb{K}[X]^{\rho\times m}$ be the matrix whose row i is $\mathbf{p}_i$. We assume that $\deg(\mathbf{P}) < D$. Then,*

1. *$\{\mathbf{p}_1, \ldots, \mathbf{p}_\rho\}$ is a $\prec$-Gröbner basis of $\mathcal{M}$ if and only if $\mathbf{P}$ is a basis of $\mathcal{M}$ such that $\mathbf{P}\pi$ is in $\mathbf{s}\pi$-weak Popov form;*

2. *$\{\mathbf{p}_1, \ldots, \mathbf{p}_\rho\}$ is the reduced $\prec$-Gröbner basis of $\mathcal{M}$ if and only if $\mathbf{P}$ is the basis of $\mathcal{M}$ such that $\mathbf{P}\pi$ is in $\mathbf{s}\pi$-Popov form, up to permutation of its rows.*

Proof. This follows from the definitions of Gröbner bases and shifted (weak) Popov forms, seeing $\prec$ as a π-permuted $\mathbf{s}$-TOP order as in Lemma 1.48, and using the link between shifted TOP orders and shifted pivots as written in Example 1.40. ☙

Concerning the first item, we note that any $\prec$-Gröbner basis consisting of ρ elements is a minimal $\prec$-Gröbner basis, up to making the $\prec$-leading terms of these elements monic.

1.3.5 Modules of finite (co)dimension and multiplication matrices

We now come back to the multivariate case, with $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X_1, \ldots, X_r]$. In this document, we will often encounter $\mathbb{K}[\boldsymbol{X}]$-modules that have finite dimension or finite codimension when considered as a $\mathbb{K}$-vector space. In this section, we focus on this case and introduce some tools for computations in such modules.

Multiplication matrices. A $\mathbb{K}[\boldsymbol{X}]$-module is in particular a $\mathbb{K}$-vector space; besides, the variables induce morphisms $\varphi_k = (\mathcal{M} \to \mathcal{M}; f \mapsto X_k f)$ for $1 \leqslant k \leqslant r$ which are pairwise commuting, that is, $\varphi_j \circ \varphi_k = \varphi_k \circ \varphi_j$ for all j, k. In fact, there is a correspondence

$$\left\{ \mathcal{M} \text{ a } \mathbb{K}[X_1, \ldots, X_r]\text{-module} \right\} \quad \longleftrightarrow \quad \left\{ \begin{array}{c} \mathcal{M} \text{ a } \mathbb{K}\text{-vector space} \\ \text{and} \\ \text{pairwise commuting morphisms} \\ \varphi_k : \mathcal{M} \to \mathcal{M},\ 1 \leqslant k \leqslant r \end{array} \right\}$$

(see for example [DF04, Section 10.1] for more details in the case of one variable).

These morphisms give the action of the variables on $\mathcal{M}$ seen as a $\mathbb{K}$-vector space. Now, if $\mathcal{M}$ has finite dimension D as a $\mathbb{K}$-vector space, and up to the choice of a vector space basis $\mathcal{E} = \{\varepsilon_1, \ldots, \varepsilon_D\}$, the morphisms $\{\varphi_k, 1 \leqslant k \leqslant r\}$ can be represented by matrices $\{\mathbf{M}_k \in \mathbb{K}^{D \times D}, 1 \leqslant k \leqslant r\}$, which are therefore pairwise commuting. In what follows, we call these matrices the *multiplication matrices* of the variables $X_1, \ldots, X_r$ with respect to the module $\mathcal{M}$ and the basis $\mathcal{E}$.

As a consequence, for a polynomial $f \in \mathcal{M}$ whose coefficients on the basis $\mathcal{E}$ form a row vector $\mathbf{f} \in \mathbb{K}^{1 \times D}$, then the coefficients of $\varphi_k(f) = X_k f$ on the same basis are given by $\mathbf{f}\,\mathbf{M}_k \in \mathbb{K}^{1 \times D}$. Here is a basic example.

Example 1.50. For $\mathcal{I} = \langle X, Y \rangle$ of $\mathbb{K}[X, Y]$, the quotient $\mathcal{M} = \mathbb{K}[X, Y]/\mathcal{I}$ is isomorphic to $\mathbb{K}$, and thus has dimension $D = 1$ as a vector space. For $f \in \mathcal{M}$, we have $fX = 0$ and $fY = 0$, and hence the multiplication matrices of X and Y are both $0 \in \mathbb{K}^{1 \times 1}$. ✍

This illustrates the typical situation where $\mathcal{M}$ is the quotient $\mathcal{M} = \mathbb{K}[\boldsymbol{X}]/\mathcal{I}$ for some ideal $\mathcal{I}$, or more generally $\mathcal{M} = \mathbb{K}[\boldsymbol{X}]^m/\mathcal{N}$ for some module $\mathcal{N}$, with $\mathcal{I}$ or $\mathcal{N}$ having finite codimension as a $\mathbb{K}$-vector space. We now give more details about this case, followed by an example.

Finite codimension. For some $m \in \mathbb{Z}_{>0}$, let $\mathcal{N}$ be a $\mathbb{K}[\boldsymbol{X}]$-submodule of $\mathbb{K}[\boldsymbol{X}]^m$ of finite codimension D, that is, the module $\mathcal{M} = \mathbb{K}[\boldsymbol{X}]^m/\mathcal{N}$ has finite dimension D as a $\mathbb{K}$-vector space. Equivalently, for all $1 \leqslant k \leqslant r$ and $1 \leqslant i \leqslant m$, there exists a nonzero univariate polynomial $f \in \mathbb{K}[X_k]$ such that $f(X_k)\mathbf{c}_i \in \mathcal{N}$.

Our goal here is to summarize some basic properties about reduced Gröbner bases of $\mathcal{N}$, about the shape of the monomial bases of $\mathcal{M}$, and about the multiplication matrices in $\mathcal{M}$. For more details, the reader may for example refer to [MMM93, Section 3].

Let $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^{1 \times m}$. Then, we choose the $\prec$-monomial basis of $\mathcal{M}$ as its vector space basis (see Section 1.3.2), which we denote by $\mathcal{E} = \{\varepsilon_1, \ldots, \varepsilon_D\}$. Besides, we denote by $\mathcal{G}$ the $\prec$-reduced Gröbner basis of $\mathcal{N}$, and by $\mathbf{M}_1, \ldots, \mathbf{M}_r$ the multiplication matrices in $\mathcal{M}$ with respect to $\mathcal{E}$.

Then, we consider the set of monomials that are obtained from those in $\mathcal{E}$ by multiplication by a variable:

$$\mathcal{S} = \{X_k \varepsilon_j, 1 \leqslant k \leqslant r, 1 \leqslant j \leqslant D\} \cup \{\mathbf{c}_i, 1 \leqslant i \leqslant m \mid \mathbf{c}_i \notin \mathcal{E}\}.$$

This set is of particular interest since the $\prec$-normal forms of its monomials form the rows of the multiplication matrices. More precisely, for $1 \leqslant k \leqslant r$, the row j of $\mathbf{M}_k$ is given by the coefficients of the normal form $\mathrm{nf}_{\prec}(X_k \varepsilon_j)$ in the basis $\mathcal{E}$.

Another useful set of monomials is the *border* $\mathcal{B} = \mathcal{S} - \mathcal{E}$. It is a generating set of the monomial module $\mathrm{in}_{\prec}(\mathcal{N})$, and the polynomials $\{f - \mathrm{nf}_{\prec}(f), f \in \mathcal{B}\}$ form a canonical generating set of $\mathcal{N}$ which is commonly called a *border basis* of $\mathcal{N}$.

Included in this border is the minimal generating set of $\mathrm{in}_{\prec}(\mathcal{N})$. It actually corresponds to the set of monomials that are leading terms of polynomials in $\mathcal{G}$, which we denote by

$$\mathcal{L} = \{\mathrm{in}_{\prec}(\mathbf{g}), \mathbf{g} \in \mathcal{G}\} \subseteq \mathcal{B};$$

in particular, we have $\mathcal{G} = \{f - \mathrm{nf}_{\prec}(f), f \in \mathcal{L}\}$. From the knowledge of the monomial basis $\mathcal{E}$, one can determine $\mathcal{L}$ by first computing $\mathcal{B}$ and then removing elements from $\mathcal{B}$ until obtaining a set of irredundant monomials.

The size of these sets will play a role in our algorithms. We have the trivial upper bound $\mathrm{Card}(\mathcal{B}) \leqslant \mathrm{Card}(\mathcal{S}) \leqslant rD + m$, and thus $\mathrm{Card}(\mathcal{L}) = \mathrm{Card}(\mathcal{G}) \leqslant rD + m$ as well. In constrast, we recall that in the univariate case we always have $\mathrm{Card}(\mathcal{L}) = m$.

Example. Following Examples 1.30, 1.42 and 1.45, we know that the ideal

$$\begin{aligned}\mathcal{I} &= \{Q \in \mathbb{K}[X, Y_1, \ldots, Y_r] \mid Q(x_j, y_{j,1}, \ldots, y_{j,r}) = 0 \text{ for all } 1 \leqslant j \leqslant D\} \\ &= \langle M(X), Y_1 - L_1(X), \ldots, Y_r - L_r(X)\rangle\end{aligned}$$

is such that the $\prec_{\mathrm{lex}}$-monomial basis of $\mathcal{M} = \mathbb{K}[X, Y_1, \ldots, Y_r]/\mathcal{I}$ is $\mathcal{E} = \{1, X, \ldots, X^{D-1}\}$, for any lexicographic order $\prec_{\mathrm{lex}}$ with $X \prec_{\mathrm{lex}} Y_k$ for all k.

As a consequence, we have

$$\mathcal{S} = \{X, \ldots, X^D\} \cup \{Y_k, Y_kX, \ldots, Y_kX^{D-1}, 1 \leqslant k \leqslant r\};$$

hence the border is

$$\mathcal{B} = \{X^D\} \cup \{Y_k, Y_kX, \ldots, Y_kX^{D-1}, 1 \leqslant k \leqslant r\}$$

and the minimal generating set of $\mathrm{in}_{\prec_{\mathrm{lex}}}(\mathcal{I})$ is

$$\mathcal{L} = \{X^D\} \cup \{Y_1, \ldots, Y_r\}.$$

Then, writing the coefficients $M = M^{(0)} + M^{(1)}X + \cdots + M^{(D-1)}X^{D-1} + X^D$, the multiplication matrix of X is the companion matrix

$$\mathbf{M}_X = \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & 0 & 1 \\ -M^{(0)} & -M^{(1)} & \cdots & -M^{(D-1)} \end{bmatrix} \in \mathbb{K}^{D\times D}.$$

Furthermore, for $1 \leqslant k \leqslant r$, $\mathbf{M}_{Y_k}$ is the Krylov matrix

$$\mathbf{M}_{Y_k} = \begin{bmatrix} \mathbf{v}_k \\ \mathbf{v}_k\mathbf{M}_X \\ \mathbf{v}_k\mathbf{M}_X^2 \\ \vdots \\ \mathbf{v}_k\mathbf{M}_X^{D-1} \end{bmatrix} \in \mathbb{K}^{D\times D},$$

where $\mathbf{v}_k$ is given by the coefficients of $L_k = L_k^{(0)} + L_k^{(1)} X + \cdots + L_k^{(D-1)} X^{D-1}$ as

$$\mathbf{v}_k = \begin{bmatrix} L_k^{(0)} & L_k^{(1)} & \cdots & L_k^{(D-1)} \end{bmatrix} \in \mathbb{K}^{1\times D}.$$

We remark that we never used the points defining $\mathcal{I}$ in this process: in fact, this derivation of the multiplication matrices holds in general for an ideal of the form $\mathcal{I} = \langle M(X), Y_1 - L_1(X), \ldots, Y_r - L_r(X)\rangle$, independently of our knowing the roots of $M(X)$. Yet, using this additional knowledge we can give another vector space basis of $\mathcal{M}$ which leads to other multiplication matrices. Another basis of $\mathcal{M}$ is the Lagrange basis

$$\{L_{0,i}(X), 1 \leqslant i \leqslant D\} \quad \text{where } L_{0,i} = \prod_{j\neq i} \frac{X - x_j}{x_i - x_j}\,;$$

we recall that the points $x_1, \ldots, x_D$ are pairwise distinct. The coordinates of a polynomial $f(X, Y_1, \ldots, Y_r) \in \mathcal{M}$ on this basis are the evaluations $\{f(x_i, 0, \ldots, 0), 1 \leqslant i \leqslant D\}$. It follows that the multiplication matrix of the variable X is the diagonal

$$\mathbf{M}_X = \begin{bmatrix} x_1 & & & \\ & x_2 & & \\ & & \ddots & \\ & & & x_D \end{bmatrix} \in \mathbb{K}^{D\times D}.$$

Besides, as above, we have

$$\mathbf{M}_{Y_k} = \begin{bmatrix} \mathbf{v}_k \\ \mathbf{v}_k \mathbf{M}_X \\ \mathbf{v}_k \mathbf{M}_X^2 \\ \vdots \\ \mathbf{v}_k \mathbf{M}_X^{D-1} \end{bmatrix} \in \mathbb{K}^{D\times D}$$

for $1 \leqslant k \leqslant r$, where $\mathbf{v}_k$ is now given by the evaluations of L_k at the points $x_1, \ldots, x_D$, that is,

$$\mathbf{v}_k = \begin{bmatrix} y_{1k} & y_{2k} & \cdots & y_{Dk} \end{bmatrix} \in \mathbb{K}^{1\times D}$$

by definition of L_k.

Computing with multiplication matrices. Working in a finite-dimensional module $\mathcal{M} = \mathbb{K}[\boldsymbol{X}]^{1\times m}/\mathcal{N}$ with a known $\prec$-monomial basis $\mathcal{E} = \{\varepsilon_i, 1 \leqslant i \leqslant D\}$, one can rely on linear algebra to perform operations in $\mathcal{M}$. When it comes to multiplying polynomials in this quotient $\mathcal{N}$, one may then pre-compute some specific multiplications to speed-up further operations.

One possibility is to compute the multiplication table of $\mathcal{M}$, that is, all products $\{\varepsilon_i \varepsilon_j, 1 \leqslant i, j \leqslant D\}$ [BPR06, Section 12.2]. This table is used for example in the main results of [BSS03]. If we do not take into account some possible structure of these products, this table is represented using about D^3 field elements.

Often, like in the context of algorithms for the change of monomial order, one rather pre-computes the multiplication matrices $\mathbf{M}_1, \ldots, \mathbf{M}_r$ with respect to $X_1, \ldots, X_r$ and $\mathcal{E}$. As detailed above, the row j of $\mathbf{M}_k$ contains the coefficients of $\mathrm{nf}_{\prec}(X_k \varepsilon_j)$ in $\mathcal{E}$. Then, a dense representation of these matrices uses rD^2 field elements, which is significantly less than D^3 in the interesting cases, where it is usually considered that $r \in o(D)$.

Similarly, for our problems, we will rely on the knowledge of multiplication matrices. In univariate contexts, we will focus on the design of efficient algorithms both for a dense multiplication matrix and for structured ones, namely, when it is block-diagonal with Jordan or companion blocks. In multivariate contexts, we will always consider that the multiplication matrices are arbitrary, dense, pairwise commuting matrices. Then, we will aim at cost bounds in $\tilde{\mathcal{O}}(rD^\omega)$, which is close to the number of field elements in the representation these matrices.

2

Fast computation of relation bases

In this chapter, we first present the central problem in this thesis, which asks to compute generating sets for modules of relations, also known as syzygies, among elements of a finite-dimensional module given through its multiplication matrices. Then, we give an overview of our algorithm to solve this problem using fast linear algebra over $\mathbb{K}$, and how it can be used to perform fast change of monomial order for zero-dimensional ideals.

After this general solution, we focus on specific situations, with a single multiplication matrix which exhibits some structure. Those include the computation of Hermite-Padé approximants, of some type of vector interpolants, and more generally of solutions to systems of linear modular equations over $\mathbb{K}[X]$. We detail these problems, previous work on efficient algorithms to solve them, and our contributions. These algorithms exploit the particularities of the multiplication matrix to design divide-and-conquer strategies which allow us to take advantage of fast arithmetic in $\mathbb{K}[X]$; at the base case of the recursion, we rely on our solution to the general case based on fast dense linear algebra.

Some of the consequences of the results presented in this chapter will be detailed in Chapter 3, concerning algorithms for multivariate interpolation and list-decoding (Section 3.1), and for computing normal forms of polynomial matrices (Section 3.2).

2.1 Relations or syzygies in finite-dimensional modules

In this section, we first introduce the notion of relations between elements of a module, also known as syzygies. Here, we focus on the case of a module which is finite-dimensional as a vector space. Then, we detail our framework for the problem of computing such relations, namely, we discuss the specification of the module in input of the problem. Finally, we give a short summary of our contributions on fast algorithms to solve this problem in general and particular cases. A more detailed overview of these contributions is given in Sections 2.2 to 2.5, while the full algorithms and proofs can be found in Parts II to IV.

2.1.1 Gröbner bases of multivariate modules of relations

Syzygy modules. Here, as an introduction to our main problem, we discuss the notion of syzygies and syzygy modules. For a detailed exposition the reader may refer to [Eis05];

in this reference, we find the following comment.

> "The use of 'syzygy' in this context seems to go back to Sylvester [Syl53]. The word entered the language of modern science in the seventeenth century, with the same astronomical meaning it had in ancient Greek: the conjunction or opposition of heavenly bodies."

The Sylvester matrix was also introduced in [Syl53]; a generalized form of this matrix will be central in our algorithm to compute syzygies.

Let $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X_1, \ldots, X_r]$ and let $\mathcal{M}$ be a $\mathbb{K}[\boldsymbol{X}]$-module. We are interested in the computation of relations between elements of $\mathcal{M}$, also known as *syzygies*. Formally, given some elements $f_1, \ldots, f_m \in \mathcal{M}$ we define

$$\mathrm{Syz}_{\mathcal{M}}(f_1, \ldots, f_m) = \{(p_1, \ldots, p_m) \in \mathbb{K}[\boldsymbol{X}]^m \mid p_1 f_1 + \cdots + p_m f_m = 0\};$$

note that these relations hold in $\mathcal{M}$. This set is a $\mathbb{K}[\boldsymbol{X}]$-submodule of $\mathbb{K}[\boldsymbol{X}]^m$ which is called the syzygy module of $\{f_1, \ldots, f_m\}$. Equivalently, $\mathrm{Syz}_{\mathcal{M}}(f_1, \ldots, f_m)$ can be defined as the kernel of the module morphism

$$\begin{array}{rccl} \varphi : & \mathbb{K}[\boldsymbol{X}]^m & \to & \mathcal{M} \\ & (p_1, \ldots, p_m) & \mapsto & p_1 f_1 + \cdots + p_m f_m. \end{array}$$

As a consequence, the quotient $\mathbb{K}[\boldsymbol{X}]^m / \mathrm{Syz}_{\mathcal{M}}(f_1, \ldots, f_m)$ is isomorphic to $\varphi(\mathbb{K}[\boldsymbol{X}]^m)$.

Our main problem is to compute generating sets of such modules of relations. We first give it in an abstract form, voluntarily overlooking the details of how the input module and polynomials are represented so as to cover the whole generality of the question. We will then discuss these details below.

Problem 1 – *Gröbner basis of a syzygy module*

Input:

- a $\mathbb{K}[X_1, \ldots, X_r]$-module $\mathcal{M}$,
- elements $f_1, \ldots, f_m \in \mathcal{M}$,
- a monomial order $\prec$ on $\mathbb{K}[X_1, \ldots, X_r]^m$.

Output:

- a $\prec$-Gröbner basis of $\mathrm{Syz}_{\mathcal{M}}(f_1, \ldots, f_m)$.

To highlight some important specific cases of this problem, let us come back to some examples that we encountered in the previous chapter.

Example 2.1. We start with univariate polynomials. For some $n \in \mathbb{Z}_{>0}$, let $\mathcal{M} = \mathbb{K}[X]^n$ and consider elements $f_i = (f_{i,1}, \ldots, f_{i,n}) \in \mathcal{M}$ for $1 \leqslant i \leqslant m$. Then, $\mathrm{Syz}_{\mathcal{M}}(f_1, \ldots, f_m)$ is precisely the kernel of the polynomial matrix $\mathbf{F} = [f_{ij}]_{1 \leqslant i \leqslant m, 1 \leqslant j \leqslant n} \in \mathbb{K}[X]^{m \times n}$. ✍

Example 2.2. Concerning Hermite-Padé approximation as in Example 1.5, we have $\mathcal{M} = \mathbb{K}[X]/\langle X^D \rangle$ and polynomials $(f_1, \ldots, f_m) \in \mathcal{M}^m$, and we look for relations in

$$\mathrm{Syz}_{\mathcal{M}}(f_1, \ldots, f_m) = \{(p_1, \ldots, p_m) \in \mathbb{K}[X]^m \mid p_1 f_1 + \cdots + p_m f_m = 0\}.$$

More generally, in the preamble we considered the same relations in $\mathcal{M} = \mathbb{K}[X]/\langle \mathfrak{m} \rangle$, that is, modulo an arbitrary nonzero polynomial $\mathfrak{m} \in \mathbb{K}[X]$. ✍

Example 2.3. Now, we turn to the multivariate polynomial ring $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X_1, \ldots, X_r]$. Let $\mathcal{I}$ be a zero-dimensional ideal in $\mathbb{K}[\boldsymbol{X}]$ and $\mathcal{M} = \mathbb{K}[\boldsymbol{X}]/\mathcal{I}$. Then, taking $m = 1$ and $f_1 = 1 \in \mathcal{M}$, we have

$$\mathrm{Syz}_{\mathcal{M}}(1) = \{p \in \mathbb{K}[\boldsymbol{X}] \mid p\,1 = 0\} = \{p \in \mathbb{K}[\boldsymbol{X}] \mid p \in \mathcal{I}\} = \mathcal{I}.$$

Thus, Problem 1 includes the problem of computing a $\prec$-Gröbner basis of $\mathcal{I}$, having as input a convenient representation of $\mathbb{K}[\boldsymbol{X}]/\mathcal{I}$ and a monomial order $\prec$.

In Example 1.32, we were given $\mathcal{M}$ as above and an element $g \in \mathcal{M}$. We then considered the module

$$\{(p_1, p_2) \in \mathbb{K}[\boldsymbol{X}]^2 \mid p_1 + p_2 g = 0\},$$

which is $\mathrm{Syz}_{\mathcal{M}}(1, g)$. ✍

In the last two examples, we work with an input module $\mathcal{M}$ which is finite-dimensional as a $\mathbb{K}$-vector space. Then, as explained in Section 1.3.5, it is customary to adopt linear algebra point of view via the use of multiplication matrices. This leads us to specifying Problem 1 with these matrices known as an input; this is the central problem in this thesis, which we detail in the next paragraph.

Relations in finite-dimensional modules. In what follows, we assume that $\mathcal{M}$ has finite dimension D as a $\mathbb{K}$-vector space. Then, choosing a vector space basis $\mathcal{E}$ of $\mathcal{M}$ allows us to identify $\mathcal{M}$ with $\mathbb{K}^D$. Furthermore, we denote by $\mathbf{M}_1, \ldots, \mathbf{M}_r$ the multiplication matrices with respect to this basis $\mathcal{E}$; we recall that they are pairwise commuting.

From the discussion in Section 1.3.5, the data formed by this basis and these matrices completely define the input module $\mathcal{M}$ of Problem 1, and thus can be used to represent it. Indeed, the module structure of $\mathcal{M}$ is as follows: for $p \in \mathbb{K}[\boldsymbol{X}]$, and for $f \in \mathcal{M}$ represented by the vector $\mathbf{f} \in \mathbb{K}^{1\times D}$ of its coefficients on the basis $\mathcal{E}$, the coefficients of $pf \in \mathcal{M}$ on the basis $\mathcal{E}$ are $\mathbf{f}\,p(\mathbf{M}_1, \ldots, \mathbf{M}_r) \in \mathbb{K}^{1\times D}$; in what follows, this vector is denoted by $p \cdot \mathbf{f}$.

Following this $\mathbb{K}$-linear algebra point of view, we will assume that the input elements $f_1, \ldots, f_m \in \mathcal{M}$ are known through the vectors $\mathbf{f}_1, \ldots, \mathbf{f}_m \in \mathbb{K}^{1\times D}$ of their coefficients on the basis $\mathcal{E}$. In this context, syzygy modules can be described as follows.

Definition 2.4. *For some $D \in \mathbb{Z}_{>0}$, let $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$ be pairwise commuting matrices in $\mathbb{K}^{D\times D}$, and let $\mathbf{F} \in \mathbb{K}^{m\times D}$. Denoting by $\mathbf{F}_{1,*}, \ldots, \mathbf{F}_{m,*}$ the rows of $\mathbf{F}$, for any polynomial $\mathbf{p} = (p_1, \ldots, p_m) \in \mathbb{K}[\boldsymbol{X}]^{1\times m}$ we write $\mathbf{p} \cdot \mathbf{F} = p_1 \cdot \mathbf{F}_{1,*} + \cdots + p_m \cdot \mathbf{F}_{m,*} \in \mathbb{K}^{1\times D}$. Then, we define the set*

$$\mathrm{Syz}_{\mathbf{M}}(\mathbf{F}) = \{\mathbf{p} \in \mathbb{K}[\boldsymbol{X}]^{1\times m} \mid \mathbf{p} \cdot \mathbf{F} = 0\}, \tag{2.1}$$

whose elements are called relations of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

The correspondence between $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ and $\mathrm{Syz}_{\mathcal{M}}(f_1, \ldots, f_m)$ directly follows from the discussion above. Furthermore, we have seen that the quotient $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\mathrm{Syz}_{\mathcal{M}}(f_1, \ldots, f_m)$ is isomorphic to a submodule of $\mathcal{M}$, hence the next result.

Lemma 2.5. $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ *is a* $\mathbb{K}[\boldsymbol{X}]$*-submodule of* $\mathbb{K}[\boldsymbol{X}]^{1\times m}$*, and the dimension of the quotient module* $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ *as a* $\mathbb{K}$*-vector space is at most* D*.*

Remark 2.6. Concerning terminology, in the specific case of $m = 1$, one would equivalently say that the ideal $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ is *zero-dimensional* and of *degree* at most D. ☕

In many situations, one wants to compute a generating set of this module which has good properties, such as a Gröbner basis. For a given monomial order $\prec$ over $\mathbb{K}[\boldsymbol{X}]^{1\times m}$, a set of polynomials in $\mathbb{K}[\boldsymbol{X}]^{1\times m}$ is said to be a $\prec$*-Gröbner relation basis of* $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ if it is a $\prec$-Gröbner basis of the module $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

The central problem in this thesis, in the multivariate case, is the following.

Problem 2 – *Gröbner relation basis*

Input:

- pairwise commuting matrices $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$ in $\mathbb{K}^{D\times D}$,
- a matrix $\mathbf{F} \in \mathbb{K}^{m\times D}$,
- a monomial order $\prec$ on $\mathbb{K}[X_1, \ldots, X_r]^m$.

Output:

- a $\prec$-Gröbner relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

To reduce an instance of Problem 1 for a finite-dimensional module $\mathcal{M}$ to an instance of Problem 2, one needs to compute a vector space basis of $\mathcal{M}$, the corresponding multiplication matrices, and the coefficients of the input elements of $\mathcal{M}$ on this basis.

For an example where the multiplication matrices are not known a priori, one may consider the problem of change of monomial order. Interestingly, the most efficient algorithms for this problem do compute the multiplication matrices and use them to obtain the sought Gröbner basis as a solution to an instance of Problem 2 [FGLM93, FGHR13]. We give more details about this specific problem in Section 2.2.3 and Chapter 5.

Interpolation and relations in a module given by a dual basis. Another situation where the multiplication matrices are not known a priori is when considering multivariate interpolation as in Example 1.30, and more generally, when the module $\mathcal{M}$ is described by a dual basis. We give below the basic tools that we will use concerning this duality; more details can be found in [MMM93] and [Mor09].

Let $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X_1, \ldots, X_r]$ and $n \in \mathbb{Z}_{>0}$. In what follows, we call *linear functional on* $\mathbb{K}[\boldsymbol{X}]^n$ any morphism $\ell : \mathbb{K}[\boldsymbol{X}]^n \to \mathbb{K}$ of $\mathbb{K}$-vector spaces. Then, let

$$\{\ell_i : \mathbb{K}[\boldsymbol{X}]^n \to \mathbb{K},\ 1 \leqslant i \leqslant D\}$$

be a set of such linear functionals that are furthermore $\mathbb{K}$-linearly independent. Defining the morphism

$$\begin{array}{rccc} \psi : & \mathbb{K}[\boldsymbol{X}]^n & \to & \mathbb{K}^D \\ & f & \mapsto & (\ell_1(f), \ldots, \ell_D(f)), \end{array}$$

of $\mathbb{K}$-vector spaces, we denote its kernel by

$$\ker(\psi) = \bigcap_{1 \leqslant i \leqslant D} \ker(\ell_i(f)).$$

Then, the quotient $\mathbb{K}[\boldsymbol{X}]^n / \ker(\psi)$ is isomorphic to $\psi(\mathbb{K}[\boldsymbol{X}]^n) = \mathbb{K}^D$, so that $\ker(\psi)$ is a $\mathbb{K}$-vector space of finite codimension D. However $\ker(\psi)$ is not a $\mathbb{K}[\boldsymbol{X}]$-submodule of $\mathbb{K}[\boldsymbol{X}]^n$ in general, and we have the following characterization:

$$\begin{aligned} &\ker(\psi) \text{ is a } \mathbb{K}[\boldsymbol{X}]\text{-module} \\ \iff &\text{ for all } i, k, \text{the functional } f \mapsto \ell_i(X_k f) \text{ is a } \mathbb{K}\text{-linear combination of } \{\ell_1, \ldots, \ell_D\}. \end{aligned}$$

For a proof, we refer the reader to [MMM93, Proposition 1.3]. Thus, in this case, we can consider the module $\mathcal{M} = \mathbb{K}[\boldsymbol{X}]^n / \ker(\psi)$; as a vector space, it has dimension D and admits $\{\ell_1, \ldots, \ell_D\}$ as a basis of its dual.

Having a description of $\mathcal{M}$ provided by $\{\ell_1, \ldots, \ell_D\}$, we may consider the computation of Gröbner bases of syzygy modules as in Problem 1. Here, to match this representation of $\mathcal{M}$, the input elements $f_1, \ldots, f_m \in \mathcal{M}$ are given through their evaluations $\{\psi(f_1), \ldots, \psi(f_m)\}$ by the linear functionals, which form m vectors in $\mathbb{K}^D$. This specific case of Problem 1 is summarized in Problem 3.

Problem 3 – *Gröbner basis of a syzygy module defined by a dual basis*

Input:

- linear functionals $\{\ell_i : \mathbb{K}[X_1, \ldots, X_r]^n \to \mathbb{K},\ 1 \leqslant i \leqslant D\}$,
- vectors $\mathbf{f}_1, \ldots, \mathbf{f}_m$ in $\mathbb{K}^D$,
- a monomial order $\prec$ on $\mathbb{K}[X_1, \ldots, X_r]^m$.

Assumptions:

- $\{\ell_1, \ldots, \ell_D\}$ are $\mathbb{K}$-linearly independent,
- for all i and k, $f \mapsto \ell_i(X_k f)$ is a $\mathbb{K}$-linear combination of $\{\ell_1, \ldots, \ell_D\}$.

Output:

- a $\prec$-Gröbner basis of the submodule

$$\mathrm{Syz}_{\mathcal{M}}(f_1, \ldots, f_m) \subseteq \mathbb{K}[X_1, \ldots, X_r]^m,$$

  where f_i is the unique element of $\mathbb{K}[X_1, \ldots, X_r]^n / \cap_{1 \leqslant j \leqslant D} \ker(\ell_j)$ such that $\mathbf{f}_i = (\ell_1(f_i), \ldots, \ell_D(f_i))$.

This problem can be reduced to Problem 2 via the computation of the multiplication matrices for $\mathcal{M}$. In this context, these can be described as follows: the multiplication matrix $\mathbf{M}_k$ for X_k is the matrix in $\mathbb{K}^{D \times D}$ whose row $i \in \{1, \ldots, D\}$ is formed by the coefficients of the linear combination which expresses $f \mapsto \ell_i(X_k f)$ on the basis $\{\ell_1, \ldots, \ell_D\}$.

The cost of this reduction from Problem 3 to Problem 2 will depend on how efficiently we can compute these multiplication matrices. As we have seen in Section 1.3.5, in some situations we have explicit formulas which allow fast computation of these matrices.

2.1.2 Univariate case: minimal relation bases

Here, we specialize the above notions to the case of univariate modules and the associated terminology. For $D \in \mathbb{Z}_{>0}$, having a matrix $\mathbf{M} \in \mathbb{K}^{D\times D}$ allows us to define a $\mathbb{K}[X]$-module structure on $\mathcal{M} = \mathbb{K}^{1\times D}$ by $p \cdot \mathbf{f} = \mathbf{f}\, p(\mathbf{M})$, for $p \in \mathbb{K}[X]$ and $\mathbf{f} \in \mathcal{M}$. As above, for $\mathbf{F} \in \mathbb{K}^{m\times D}$ and a row vector $\mathbf{p} = [p_1, \ldots, p_m] \in \mathbb{K}[X]^{1\times m}$, we write $\mathbf{p} \cdot \mathbf{F} = p_1 \cdot \mathbf{F}_{1,*} + \cdots + p_m \cdot \mathbf{F}_{m,*} \in \mathbb{K}^{1\times D}$. If $\mathbf{p} \cdot \mathbf{F} = 0$, then we say that $\mathbf{p}$ is a relation of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

This general notion of univariate relations was studied in [BL00] in the context of *rational interpolation* (for computational reasons, in this reference it is required that $\mathbf{M}$ be upper triangular). Let us now show how some examples in the previous chapter cast into this setting.

Example 2.7. In Example 1.5, this is straightforward: $\mathbf{F}$ contains the coefficients of the input polynomials $f_1, \ldots, f_m$, while the multiplication matrix is the upper shift matrix

$$\mathbf{Z} = \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & 0 & 1 \\ & & & 0 \end{bmatrix}, \tag{2.2}$$

which is a nilpotent Jordan block. More generally, if the modulus X^D in this example is replaced by an arbitrary monic polynomial $\mathfrak{m} = c_0 + c_1 X + \cdots + c_{D-1} X^{D-1} + X^D$, then $\mathbf{F}$ is built similarly while the multiplication matrix is the companion matrix

$$\mathbf{C} = \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & 0 & 1 \\ -c_0 & -c_1 & \cdots & -c_{D-1} \end{bmatrix}.$$

Now, note that Example 1.4 is a bivariate problem that has been linearized into a problem in $\mathbb{K}[X]^{1\times m}$ by requiring that the solutions satisfy $\deg_Y(Q) < m$. Here, we consider the diagonal multiplication matrix

$$\mathbf{M} = \begin{bmatrix} x_1 & & & \\ & x_2 & & \\ & & \ddots & \\ & & & x_D \end{bmatrix}.$$

Then, for a polynomial $p \in \mathbb{K}[X]$ and a vector $\mathbf{f} = [e_1, \ldots, e_D] \in \mathcal{M}$ which is thought of as the evaluations of some bivariate polynomial at the points $\{(x_1, y_1), \ldots, (x_D, y_D)\}$, we have $p \cdot \mathbf{f} = [p(x_1)e_1, \ldots, p(x_D)e_D]$. In this case, to solve the bivariate interpolation problem, we start from the tuple of bivariate polynomials $(1, Y, \ldots, Y^{m-1})$; their evaluations $\mathbf{F} = (\mathbf{f}_1, \ldots, \mathbf{f}_m)$ in $\mathcal{M}^m$ are the vectors $\mathbf{f}_i = [y_1^i, \ldots, y_D^i]$. Then, for $\mathbf{p} = [p_1, \ldots, p_m]$, the relation $\mathbf{p} \cdot \mathbf{F} = 0$ precisely means that $Q(X, Y) = p_1 + p_2 Y + \cdots + p_m Y^{m-1}$ vanishes at all points $\{(x_1, y_1), \ldots, (x_D, y_D)\}$. ✍

The main results in this thesis about the computation of univariate relations deal with several situations which generalize these examples; each situation gives access to different techniques and leads to different algorithms. The first one holds in the general case with no assumption on $\mathbf{M}$; it is used in particular in all other situations when D is small with respect to m, thus providing a fast unified solution in this case. The second result requires that $\mathbf{M}$ be block-diagonal with companion blocks, which roughly amounts to knowing the invariant factors of the module $\mathcal{M}$. The third one assumes that $\mathbf{M}$ is a Jordan matrix, and the fourth one that $\mathbf{M}$ is a nilpotent Jordan matrix; these both mean that the invariant factors of $\mathcal{M}$ split over $\mathbb{K}$ and that we further know their roots and multiplicities.

Let us come back to our general context. As we have seen above, for $\mathbf{M} \in \mathbb{K}^{D\times D}$ and $\mathbf{F} \in \mathbb{K}^{m\times D}$, the set of relations

$$\mathrm{Syz}_{\mathbf{M}}(\mathbf{F}) = \{\mathbf{p} \in \mathbb{K}[X]^{1\times m} \mid \mathbf{p}\cdot\mathbf{F} = 0\}$$

is a $\mathbb{K}[X]$-submodule of $\mathbb{K}[X]^{1\times m}$. Furthermore, $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ contains $\pi_{\mathbf{M}}(X)\mathbb{K}[X]^{1\times m}$, where $\pi_{\mathbf{M}} \in \mathbb{K}[X]$ is any nonzero polynomial which annihilates $\mathbf{M}$. Then, $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ is free of rank m according to Lemma 1.1, hence the following definition.

Definition 2.8 (Relation basis)**.** *For $\mathbf{M} \in \mathbb{K}^{D\times D}$ and $\mathbf{F} \in \mathbb{K}^{m\times D}$, a matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ is a* relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ *if its rows form a basis of the $\mathbb{K}[X]$-module $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.*

In terms of notation, if a matrix $\mathbf{P} \in \mathbb{K}[X]^{k\times m}$ has rows $\mathbf{p}_1, \ldots, \mathbf{p}_k$, we write $\mathbf{P}\cdot\mathbf{F}$ for the matrix in $\mathbb{K}^{k\times D}$ whose rows are $\mathbf{p}_1\cdot\mathbf{F}, \ldots, \mathbf{p}_k\cdot\mathbf{F}$. In particular, if $k = m$ and $\mathbf{P}$ is a relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$, then $\mathbf{P}\cdot\mathbf{F} = 0$.

In many situations, one wants to compute a relation with degree constraints; in fact, as we will observe in the next sections when discussing previous work, most known algorithms for finding univariate relations compute a relation basis in shifted reduced form.

Definition 2.9 (Shifted minimal relation basis)**.** *Let $\mathbf{M} \in \mathbb{K}^{D\times D}$, $\mathbf{F} \in \mathbb{K}^{m\times D}$, and $\mathbf{s} \in \mathbb{Z}^m$. Then, a matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ is said to be an* $\mathbf{s}$-minimal relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ *if*

- $\mathbf{P}$ *is a relation basis of* $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$*, and*
- $\mathbf{P}$ *is* $\mathbf{s}$*-reduced.*

From Chapter 1, we know that a row of $\mathbf{P}$ of minimal $\mathbf{s}$-row degree also has minimal $\mathbf{s}$-row degree among *all* relations of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$; furthermore, there is a unique relation basis which is in $\mathbf{s}$-Popov form, called the $\mathbf{s}$-*Popov relation basis of* $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

This leads us to Problem 4, which is the central problem of this thesis in univariate contexts.

We state here a preliminary result which is at the core of the cost analyses of algorithms solving Problem 4. It gives the bound D on the degree of the determinant of any relation basis, which as explained in Section 1.2.2 also provides us with bounds for average row (resp. column) degrees of shifted minimal (resp. Popov) relation bases.

Lemma 2.10. *Let $\mathbf{M} \in \mathbb{K}^{D\times D}$, let $\mathbf{F} \in \mathcal{M}^m$, and let $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ be a relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$. Then, $\deg(\det(\mathbf{P})) \leqslant D$.*

Proof. From the discussion following Definition 1.22, we know that $\deg(\det(\mathbf{P}))$ is the dimension of $\mathbb{K}[X]^{1\times m}/\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ as a $\mathbb{K}$-vector space. According to Lemma 2.5, this dimension is at most D. ∎

Problem 4 – *Minimal relation basis*

Input:

- a matrix $\mathbf{M} \in \mathbb{K}^{D \times D}$,
- a matrix $\mathbf{F} \in \mathbb{K}^{m \times D}$,
- a shift $\mathbf{s} \in \mathbb{Z}^m$.

Output:

- an $\mathbf{s}$-minimal relation basis $\mathbf{P} \in \mathbb{K}[X]^{m \times m}$ for $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

2.1.3 Overview of our results

We present in Section 2.2 our algorithm to deal with the most general case of our main problem: Problem 2 with arbitrary multiplication matrices and monomial order. Detailed in Chapter 4, this algorithm relies on a complete linearization of the problem, the main computational task being to find the row rank profile of some structured matrix called a *multi-Krylov matrix*. Exploiting several techniques from fast linear algebra over $\mathbb{K}$, it computes the reduced Gröbner relation basis in $\tilde{\mathcal{O}}(rD^\omega)$ operations.

As a particular case, when working with univariate polynomials, this algorithm computes the shifted Popov relation basis in $\tilde{\mathcal{O}}(D^\omega)$ field operations for an arbitrary multiplication matrix and an arbitrary shift. In many univariate contexts, one has information on the structure of the input module, which translates as $\mathbf{M}$ having some particular structure, such as these observed in Example 2.7. Then, focusing on the univariate case, we will present in Sections 2.3 to 2.5 algorithms that solve Problem 4 for several families of multiplication matrices $\mathbf{M}$. These algorithms all rely on the general algorithm in $\tilde{\mathcal{O}}(D^\omega)$ for the base cases of their recursion.

Concerning the case of a *nilpotent Jordan matrix* $\mathbf{M}$ (Section 2.4 and Chapter 7), the corresponding relation bases are also known as approximant bases or order bases. In this context, a product of the form $\mathbf{P} \cdot \mathbf{F}$ is simply a product of polynomial matrices. We obtain the cost bound $\tilde{\mathcal{O}}(m^{\omega-1}D)$ for computing the shifted Popov basis for arbitrary shifts, while this cost was previously obtained under restrictive assumptions on the shift and for returning a minimal basis which was not normalized. This improvement is based on the strategy mentioned in Section 1.2.1: we first recursively find information on the degrees in the shifted Popov basis, and then use this additional knowledge to rely on fast algorithms specialized for almost uniform shifts.

Then, we turn to the case of a *Jordan matrix* $\mathbf{M}$ (Section 2.4 and Chapter 14). For this specific triangular form, which can be represented by $\mathcal{O}(D)$ field elements, we manage to design an efficient algorithm for computing products $\mathbf{P} \cdot \mathbf{F}$. Using the above-mentioned divide-and-conquer approach along with a new ingredient to control the degrees in the recursion, namely a fast change of shift, we obtain an algorithm in $\tilde{\mathcal{O}}(m^{\omega-1}D)$ to compute a minimal basis for almost uniform shifts. Building on this, we obtain a similarly fast algorithm which returns the shifted Popov basis for an arbitrary shift.

Finally, we study the case of a *companion-block diagonal* matrix $\mathbf{M}$ (Section 2.5 and Chapter 8), which has only $\mathcal{O}(D)$ nonzero coefficients in $\mathbb{K}$ as well. We will relate this to the computation of bases of solutions to systems of linear modular equations;

as such, this situation generalizes the previous case of a Jordan matrix. We also obtain the cost bound $\tilde{\mathcal{O}}(m^{\omega-1}D)$ for an arbitrary shift and for returning the shifted Popov basis, under the assumption that the number of blocks is in $\mathcal{O}(m)$; furthermore, this assumption implies that there is a simple and efficient algorithm to compute products of the form $\mathbf{P} \cdot \mathbf{F}$. However, since $\mathbf{M}$ is not triangular we do not have recurrence equations that would lead to an iterative algorithm such as those in [Bec92, VBB92, BL94, BL00]. Then, relying on kernel basis and approximant basis computations is a natural approach that we use in our algorithm, while also introducing a new ingredient to handle arbitrary shifts.

Remark 2.11. More generally, having $\mathbf{M}$ upper triangular like in [BL00] leads to a situation which bears computational similarities to that of a nilpotent Jordan matrix. Precisely, it provides us with recurrence equations [BL00, Theorem 6.1] which yield an iterative algorithm [BL00, Algorithm FFFG], recalled here in Section 6.4. This iteration can be turned into a divide-and-conquer algorithm, for example using the leading and trailing principal submatrices of $\mathbf{M}$ for the recursive calls.

For approximation problems, such iterative algorithms have been given in [Bec92, VBB92, BL94], and more specifically for approximant basis computation such divide-and-conquer algorithms can be found in [BL94, GJV03]. Yet, for an arbitrary triangular $\mathbf{M}$, we cannot obtain an algorithm with cost quasi-linear in D as simply representing $\mathbf{M}$ requires $\Theta(D^2)$ coefficients in $\mathbb{K}$; in this case, the fastest known approach is our algorithm in $\tilde{\mathcal{O}}(D^\omega)$ for a dense matrix $\mathbf{M}$, thus ignoring its triangular shape. ☕

2.2 Fast algorithms for dense multiplication matrices

Here, we give a detailed overview of our most general result, concerning the computation of multivariate relation bases (Problem 2). In this case, for r variables and an input module of vector space dimension D, we obtain a deterministic algorithm whose cost is essentially that of performing fast linear algebra operations with r scalar matrices of dimensions $D \times D$. Here, we detail this result in the univariate and multivariate cases, we compare it to previous work, and we give an overview of our algorithm.

2.2.1 Results

Univariate case. We start with the case of one variable, that is, Problem 4. Here, we have a single multiplication matrix $\mathbf{M} \in \mathbb{K}^{D \times D}$. Our algorithm performs an iteration on the degree of the sought relations, up to a specified degree bound β that we discuss in the next paragraph. In our framework, multiplication of a scalar vector by X corresponds to its multiplication by $\mathbf{M}$, and as a consequence the main computational task in our algorithm is to compute powers of $\mathbf{M}$. With an idea already used in [KG85], we restrict to computing powers of the form $\mathbf{M}^{2^e}$, for e between 0 and $\log_2(\min(\beta, \Delta + 1))$.

This number of steps is parameterized by two integers. First, $\beta \in \mathbb{Z}_{>0}$ is a user-chosen bound on the degree of the sought $\mathbf{s}$-Popov relation basis $\mathbf{P}$ for $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$. If we denote by $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m)$ the diagonal degrees of $\mathbf{P}$, we equivalently require that $\beta > \max(\boldsymbol{\delta})$.

Remark however that $\boldsymbol{\delta}$ is unknown a priori; if no context-specific bound β is known, one may always choose $\beta = D + 1$ since

$$\max(\boldsymbol{\delta}) \leqslant \delta_1 + \cdots + \delta_m = \deg(\det(\mathbf{P})) \leqslant D$$

according to Lemma 2.10. Besides, any bound on the degree of the minimal polynomial of $\mathbf{M}$ also provides a valid bound β.

The second parameter is the quantity $\Delta = \delta_1 + \cdots + \delta_m$, which is the dimension of $\mathbb{K}[X]^{1\times m}/\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$ as a $\mathbb{K}$-vector space (see Section 1.1.3) and is unknown a priori. If no good bound β is known, the algorithm will still stop after dealing with the power Δ of $\mathbf{M}$ by detecting that no relations of higher degree will be found, which is the case since $\deg(\mathbf{P}) \leqslant \deg(\det(\mathbf{P})) = \Delta$.

The next result follows from Section 4.3.2, and in particular Algorithm 3.

Theorem 2.12. *Let $\mathbf{M} \in \mathbb{K}^{D\times D}$, $\mathbf{F} \in \mathbb{K}^{m\times D}$, and $\mathbf{s} \in \mathbb{Z}^m$. Let $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ be the $\mathbf{s}$-Popov relation basis of $\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$, and let $\Delta = \deg(\det(\mathbf{P}))$. Suppose that a bound $\beta \in \mathbb{Z}_{>0}$ such that $\beta > \deg(\mathbf{P})$ is known a priori; by default, one may choose $\beta = D + 1$. Then, there is a deterministic algorithm which solves Problem 4 using*

$$\begin{aligned}&\mathcal{O}(mD^{\omega-1} + D^{\omega}\log(\min(\beta, \Delta + 1)))\\ &\subseteq \mathcal{O}(mD^{\omega-1} + D^{\omega}\log(D))\end{aligned}$$

operations in $\mathbb{K}$ and returns the $\mathbf{s}$-Popov basis $\mathbf{P}$.

This cost can be compared to the size of the manipulated objects: here, representing the input of the problem uses $\Theta(mD+D^2)$ field elements. It may however strike the reader that when $D = \mathcal{O}(1)$, the cost bound above is linear in m while the dense representation of the output $m \times m$ polynomial matrix will use at least m^2 field elements. To explain this, we refer to Section 4.3: we will see that when $D < m$, at least $m - D$ columns of the basis in $\mathbf{s}$-Popov form are coordinate column vectors with 1 on the diagonal. In the algorithm, these columns are described without involving any arithmetic operation, and hence the actual computation is restricted to an $m \times D$ submatrix of the output basis.

In this thesis, we will often use this result for the base case of our recursive algorithms for multiplication matrices that have specific structures. In this situation, the dimension D will be between $m/2$ and m, and therefore the cost bound above becomes $\mathcal{O}(m^{\omega}\log(m))$.

An overview of results related to Problem 4 can be found in [Kai80, Chapter 6]; algorithms based on Gaussian elimination are presented for the Hermite and Popov forms. In [Vil96, Section 4], the Hermite form of a matrix in Popov form is computed by relying on an instance of Problem 4, using ideas from [KG85] for more efficiency.

In the context of rational interpolation, Problem 4 was considered for arbitrary shifts in [BL00], focusing on the design of a fraction-free algorithm for an upper triangular multiplication matrix $\mathbf{M}$. As explained in Remark 2.11, this assumption implies that one can use recurrence equations which leads to find the shifted Popov relation basis via an iteration on the columns of $\mathbf{F}$ [BL00, Algorithm FFFG].

In Section 6.4, we present the latter algorithm rewritten with our notation, and we observe that for an arbitrary triangular matrix, it uses $\mathcal{O}(mD^2+D^3)$ field operations. As a comparison, our algorithm supporting Theorem 2.12 is faster as soon as one uses subcubic matrix multiplication ($\omega < 3$), and works for non-triangular multiplication matrices.

Multivariate case. When working with several variables, we have several corresponding multiplication matrices $\mathbf{M}_1, \ldots, \mathbf{M}_r$. Our algorithm to solve Problem 2 deals with one multiplication matrix after another, allowing us to rely on an approach that is similar to that above for the univariate case. This also helps us to introduce fast matrix multiplication, by avoiding the computation of many vector-matrix products involving each time a different matrix $\mathbf{M}_k$, and instead grouping these operations into some matrix-matrix products involving $\mathbf{M}_1$, then some others involving $\mathbf{M}_2$, etc.

A detailed overview of our algorithm is given in Section 2.2.2. Its most expensive computational task is to repeatedly square $D \times D$ matrices. As above, the number of such squarings will depend on the dimension $\Delta \in \mathbb{Z}_{\geqslant 0}$ of the quotient $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$, as well as on degree bounds $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_r) \in \mathbb{Z}^r_{>0}$ specified by the user and which satisfy the assumption

$$\mathcal{H}_{\boldsymbol{\beta}}: \quad \beta_k > \max_{1\leqslant j\leqslant s} \deg_{X_k}(\mathbf{p}_j) \ \text{ for } 1 \leqslant k \leqslant r,$$

where $\{\mathbf{p}_1, \ldots, \mathbf{p}_s\} \subset \mathbb{K}[\boldsymbol{X}]^{1\times m}$ is the sought $\prec$-reduced Gröbner relation basis of $\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$. Since $\Delta \leqslant D$ according to Lemma 2.5, the choice $\boldsymbol{\beta} = (D+1, \ldots, D+1)$ ensures $\mathcal{H}_{\boldsymbol{\beta}}$.

The following result follows from Proposition 4.19 and the corresponding Algorithm 4.

Theorem 2.13. *Let $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$ be pairwise commuting matrices in $\mathbb{K}^{D\times D}$, let $\mathbf{F} \in \mathbb{K}^{m\times D}$, let $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^{1\times m}$, and let $\boldsymbol{\beta} \in \mathbb{Z}^r_{>0}$ be such that $\mathcal{H}_{\boldsymbol{\beta}}$; by default, one may choose $\boldsymbol{\beta} = (D+1, \ldots, D+1)$. Then, there is a deterministic algorithm which solves Problem 2 using*

$$\begin{aligned}&\mathcal{O}\big(mD^{\omega-1} + D^{\omega}\log\big(2^r \textstyle\prod_{1\leqslant k\leqslant r} \min(\beta_k, \Delta+1)\big)\big)\\ &\quad\subseteq \mathcal{O}\big(mD^{\omega-1} + rD^{\omega}\log(D)\big)\end{aligned}$$

operations in $\mathbb{K}$ and returns the $\prec$-reduced Gröbner basis. In this cost bound, $\Delta \in \mathbb{Z}_{\geqslant 0}$ stands for the dimension of $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$ as a $\mathbb{K}$-vector space.

This cost bound can be compared to the size of the representation of the input and output of the problem. The size of the input matrices $\mathbf{M}$ and $\mathbf{F}$ is in $\Theta(mD + rD^2)$. The output reduced Gröbner basis can be represented via s vectors in $\mathbb{K}^{\Delta}$, each of them giving the coefficients of the normal form of one of the minimal generators of the initial module of $\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$. Then, representing this Gröbner basis uses $\mathcal{O}(s\Delta) \subseteq \mathcal{O}(rD^2)$ field elements.

Commonly encountered situations involve $m \leqslant D$. In this case, $mD^{\omega-1} \in \mathcal{O}(D^{\omega})$ and the cost bound can be simplified as $\mathcal{O}(rD^{\omega}\log(D))$. A typical example is if one uses this result to perform a change of monomial order for an ideal, via the preliminary computation of the multiplication matrices: then, we have $m = 1$.

2.2.2 Overview of our algorithm

Linear algebra viewpoint. To introduce matrix multiplication in our solution to Problem 2, we rely on a linearization of the problem into questions of linear algebra over $\mathbb{K}$. From $\mathbf{M}$ and $\mathbf{F}$, we build a matrix over $\mathbb{K}$ whose nullspace corresponds to a set of relations of $\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$. This matrix is called a *multi-Krylov matrix*, in reference to its specific structure which exhibits a collection of Krylov subspaces of $\mathbb{K}^D$.

The multi-Krylov matrix is a generalization to several variables and to an arbitrary monomial order of the Krylov matrices, or striped-Krylov matrices, considered for example in [Kai80, BL00]. We note that its construction is similar to the Sylvester matrix and more generally to the Macaulay matrix [Syl53, Mac02, Mac16], which are commonly used when adopting a linear algebra viewpoint while dealing with operations on univariate and multivariate polynomials.

The construction of the multi-Krylov matrix is based on the multiplication $\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i \cdot \mathbf{F} = \mathbf{F}_{i,*}\mathbf{M}^{\mathbf{e}}$, for a monomial $\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i \in \mathbb{K}[\boldsymbol{X}]^{1\times m}$. Since a polynomial in $\mathbb{K}[\boldsymbol{X}]^{1\times m}$ is a $\mathbb{K}$-linear combination of monomials, this identity means that a relation of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ may be interpreted as a $\mathbb{K}$-linear relation between row vectors of the form $\mathbf{F}_{i,*}\mathbf{M}^{\mathbf{e}}$.

Choosing some degree bounds $\boldsymbol{\beta} \in \mathbb{Z}^r_{>0}$, this matrix is formed by all such rows $\mathbf{F}_{i,*}\mathbf{M}^{\mathbf{e}}$ for $1 \leqslant i \leqslant m$ and $\mathbf{0} \leqslant \mathbf{e} \leqslant \boldsymbol{\beta}$. This construction is detailed in Section 4.1, where we also show that the left nullspace of this matrix corresponds to the set of relations of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ whose degree in X_k does not exceed β_k, for $1 \leqslant k \leqslant r$.

Choosing sufficiently large bounds $\boldsymbol{\beta}$ (namely, ones that satisfy $\mathcal{H}_{\boldsymbol{\beta}}$) implies that the $\prec$-reduced Gröbner relation basis can be retrieved from specific rows in this nullspace; yet, we do not know which rows a priori. Ordering the rows in the multi-Krylov matrix according to the input monomial order $\prec$, we show in Section 4.2 that the row rank profile of this matrix corresponds to the $\prec$-monomial basis of the quotient $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

As explained in Section 1.3.5, from the monomial basis one can easily find the set of minimal generators for the initial module of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$. Being monomials, these generators thus indicate a submatrix of the multi-Krylov matrix; the left nullspace of this submatrix, computed in reduced echelon form, gives us the $\prec$-reduced Gröbner relation basis.

Previous work. In terms of computation, an immediate remark is that the number of field entries of the multi-Krylov matrix is $m\beta_1 \cdots \beta_r D \in \mathcal{O}(mD^{r+1})$, which largely exceeds our target cost. Exploiting the structure of this matrix is therefore a common thread in all efficient algorithms.

As noted above, the algorithms in [BL00] are designed for upper triangular multiplication matrices, and use this assumption to solve the problem by an iteration on the columns of $\mathbf{F}$. This situation was generalized to the case of several variables in [Fit97, OF02] by requiring specific assumptions on the input, in [Fit97, Eqn. (4.1)] and [OF02, Eqn. (5)]. These assumptions imply that one can solve the problem by finding iteratively Gröbner bases for a sequence of approximating solution modules which decrease towards the target solution module (see [Fit97, Algorithm 4.7] and [OF02, Algorithm 3.2]). It seems that such an iterative approach cannot be applied to the general case of Problem 2, where the input module has no other property than being finite-dimensional.

Previous algorithms dealing with similar problems can be found in [FGLM93, MMM93] and [Fit97, Section 2]. The algorithms of [FGLM93, Fit97] are specialized to the situation of the change of monomial order for ideals or modules, with a cost bound in $\mathcal{O}(rD^3)$. In [MMM93], several situations generalizing [MB82, FGLM93] are studied; the comparison to our result is unclear for the moment.

For the lexicographic order $\prec_{\mathrm{lex}}$, and under the strong assumption that the ideal is in Shape Position, fast matrix multiplication was exploited by [FGHR14]. This specific

situation implies that only $\mathbf{M}_r$ is needed; then, [FGHR14] gives a probabilistic algorithm to compute the $\prec_{\text{lex}}$-Gröbner relation basis within the cost bound $\mathcal{O}(D^\omega \log(D) + r\mathsf{M}(D)\log(D))$ ([FGHR14, Proposition 3]). Besides ideas from [FM11, FM17], this uses repeating squaring as in [KG85]. More details about the context of the change of monomial order are given in Section 2.2.3.

Here, we manage to incorporate fast scalar matrix multiplication without assumption on the module, and for an arbitrary order.

Overview of our algorithm. Above, we described our algorithm as a two-step strategy, where we first compute the row rank profile of the multi-Krylov matrix to obtain the monomial basis, and we then use this information to identify a specific submatrix whose left nullspace gives us the Gröbner relation basis.

Let us first focus on the computation of the row rank profile. Adapting the ideas in the algorithms of [FGLM93, MMM93, Fit97] to our framework, one obtains an approach that we summarize as follows. One iteratively considers the rows of the matrix, ordered by the monomial order $\prec$, looking for a linear relation with the previous rows by Gaussian elimination. When such a linear relation is found, the corresponding row will not be in the row rank profile and can be discarded. Now, the multi-Krylov structure further permits to discard all the rows that correspond to monomial multiples of the leading term of the discovered relation, even before computing these rows. At some point, the set of rows to be considered is exhausted, and the row rank profile is formed by the rows corresponding to an iteration that did not yield a linear relation.

Note that, in this approach, a row of the multi-Krylov matrix is computed only when arriving at the first iteration in which it is involved; it is obtained by multiplying one of the already computed rows by one of the multiplication matrices. This results in many vector-matrix products, with possibly different matrices each time: this is a first obstacle towards incorporating fast matrix multiplication. We circumvent this by introducing the variables one after another, thus seemingly not respecting the order of the rows specified by the monomial order. Yet, we manage to ensure that this order is respected in the end, by constantly inserting the rows at their right respective positions.

Then, when dealing with one variable X_k, we process successive powers $\mathbf{M}_k^{2^e}$ in the style of Keller-Gehrig's algorithm [KG85], using a logarithmic number of iterations. Our stopping criterion is either when we have reached the prescribed exponent bound β_k for X_k, or when we detect that no new relation was found in an iteration, and thus that all the remaining rows to be inserted with this variable have to be linear combinations of the rows already processed. The latter case occurs in particular when we are considering exponents 2^e beyond the dimension Δ of the monomial basis. As a consequence, the number of steps is at most $\log_2(\min(\beta_k, \Delta + 1))$, where at each step the most expensive operation is the squaring in $\mathcal{O}(D^\omega)$ to obtain $\mathbf{M}_k^{2^{e+1}}$ from $\mathbf{M}_k^{2^e}$.

As said above, we conclude by a nullspace computation. Having found the row rank profile of the multi-Krylov matrix allows us to deduce its submatrix corresponding to minimal generators of the initial module of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$. Then, the reduced Gröbner relation basis is efficiently computed as being the left nullspace in some reduced echelon form of this submatrix, which can be computed efficiently by matrix inversion and multiplication.

2.2.3 Change of monomial order for zero-dimensional ideals

The problem of change of monomial order is a specific kind of Gröbner basis computation where one already knows a Gröbner basis of the considered ideal, for another monomial order than the target one. This is formalized in Problem 5.

Problem 5 – *Change of monomial order*

Input:

- monomial orders $\prec_1$ and $\prec_2$ on $\mathbb{K}[X_1, \ldots, X_r]$,
- a $\prec_1$-reduced Gröbner basis $\{f_1, \ldots, f_s\}$ defining a zero-dimensional ideal $\mathcal{I} = \langle f_1, \ldots, f_s \rangle$ of $\mathbb{K}[X_1, \ldots, X_r]$.

Output:

- the $\prec_2$-reduced Gröbner basis of $\mathcal{I}$.

The main motivation for studying this problem is that it is a key component of the current fast algorithms for finding the solutions to a given system of multivariate polynomial equations. The polynomials defining these equations form an ideal $\mathcal{I} \subseteq \mathbb{K}[\boldsymbol{X}]$, and it is assumed that the number of solutions is finite, which means that the dimension of $\mathbb{K}[\boldsymbol{X}]/\mathcal{I}$ as a $\mathbb{K}$-vector space is finite. For solving such a system, it turns out that it is convenient to have a $\prec_{\text{lex}}$-Gröbner basis of $\mathcal{I}$. For example, in some cases it directly yields a univariate representation of the solutions [Rou99], in particular if the conditions of the Shape Lemma hold [KR00, Theorem 3.7.25].

For efficiency reasons, such a $\prec_{\text{lex}}$-Gröbner basis is usually obtained by first computing a $\prec_{\text{drl}}$-Gröbner basis and then resorting to a change of monomial order toward the lexicographic order. Concerning this problem, the fastest known algorithms [FGLM93, FM11, FGHR13, FGHR14, FM17] rely on a two-step strategy which is summarized in [FGHR13, Algorithm 2]. As a first step, one uses the known Gröbner basis to compute the multiplication matrices $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$ of the quotient $\mathbb{K}[\boldsymbol{X}]/\mathcal{I}$. Denoting by D the dimension of this quotient, these are r pairwise commuting matrices in $\mathbb{K}^{D \times D}$. Furthermore, we remark that, defining the vector $\mathbf{F} = [1 \;\; 0 \;\; \cdots \;\; 0] \in \mathbb{K}^{1 \times D}$ corresponding to the monomial 1 in $\mathbb{K}[\boldsymbol{X}]/\mathcal{I}$, then $\mathcal{I}$ is precisely the ideal of relations $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ (see Example 2.3). Thus, the second step consists in solving an instance of Problem 2 with input $\mathbf{M}$, $\mathbf{F}$, and the monomial order $\prec_{\text{lex}}$.

A first realization of this strategy lies in the FGLM algorithm [FGLM93], which solves Problem 5 in $\mathcal{O}(rD^3)$ field operations for arbitrary orders $\prec_1$ and $\prec_2$. In our algorithm for Problem 2, we have managed to incorporate fast linear algebra into the second step, with a result in Theorem 2.13 which can be summarized as follows in the situation here: once the multiplication matrices are known, one can find the $\prec_2$-reduced Gröbner basis in $\mathcal{O}(rD^\omega \log(D))$ operations; in the specific case of the lexicographic order, this is done slightly faster, in $\mathcal{O}(rD^\omega \log(D/r))$ operations (see Remark 4.15).

We now present our results about incorporating fast linear algebra into the computation of the multiplication matrices (Problem 6).

To obtain a fast algorithm, we will require that $\mathbb{K}$ be of characteristic zero and that the initial ideal $\mathcal{J} = \mathrm{in}_{\prec}(\mathcal{I})$ be Borel-fixed. We give some preliminaries about Borel-fixed

Problem 6 – *Computing the multiplication matrices*

Input:

- a monomial order $\prec$ on $\mathbb{K}[X_1, \ldots, X_r]$,
- a $\prec$-reduced Gröbner basis $\{f_1, \ldots, f_s\}$ defining a zero-dimensional ideal $\mathcal{I} = \langle f_1, \ldots, f_s \rangle$ of $\mathbb{K}[X_1, \ldots, X_r]$.

Output:

- the multiplication matrices $(\mathbf{M}_1, \ldots, \mathbf{M}_r)$ of the variables in $\mathbb{K}[X_1, \ldots, X_r]/\mathcal{I}$ with respect to its $\prec$-monomial basis.

ideals in Section 5.1; here, we simply mention a theorem of Galligo and Bayer-Stillman which implies that for an order $\prec$ which refines the degree, the generic initial ideal of $\mathcal{I}$ is Borel-fixed [Eis95, Theorem 15.20].

Theorem 2.14. *Assuming that $\mathbb{K}$ is a field of characteristic 0, and that $\mathrm{in}_{\prec}(\mathcal{I})$ is Borel-fixed, there is a deterministic algorithm which solves Problem 6 using $\mathcal{O}(rD^{\omega}\log(D))$ operations in $\mathbb{K}$.*

For more details, we refer the reader to Proposition 5.7 and Algorithm 7. This leads to the following result concerning the change of monomial order.

Corollary 2.15. *With the notations in Problem 5, assuming that $\mathbb{K}$ is a field of characteristic 0, and that $\mathrm{in}_{\prec}(\mathcal{I})$ is Borel-fixed, there is a deterministic algorithm which solves Problem 5 using $\mathcal{O}(rD^{\omega}\log(D))$ operations in $\mathbb{K}$.*

As mentioned above, for polynomial system solving an interesting particular case is that of $\prec_1$ being the degree-reverse lexicographic order and $\prec_2$ being the lexicographic order. Fast algorithms for this case have been studied in [FGHR14].

In this reference, it is assumed that the ideal $\mathcal{I}$ is in Shape Position. In this context, one does not need to compute all multiplication matrices for the change of monomial order: the matrix $\mathbf{M}_r$ suffices. A first result in [FGHR14] is that, under the Moreno-Socias conjecture [MS91, MS03a] and in the case of a generic ideal, the matrix $\mathbf{M}_r$ can be read off from the given $\prec_{\mathrm{drl}}$-Gröbner basis. Then, a second result in this reference states that this holds more generally up to a random linear change of coordinates. To prove this, [FGHR14] uses a property of Borel-fixedness of the $\prec_{\mathrm{drl}}$-initial ideal of $\mathcal{I}$ (see Section 5.1 for more details). Here, we elaborate over this to efficiently compute all the multiplication matrices when this property is satisfied; concerning the change of monomial order, this allows us to also incorporate matrix multiplication for ideals which are not in Shape Position.

In [FGHR13], an algorithm is designed to compute the multiplication matrices from a $\prec_{\mathrm{drl}}$-Gröbner basis in $\mathcal{O}(\beta r^{\omega} D^{\omega})$, where β is the maximum total degree of the elements of the input Gröbner basis. This is obtained by iterating over the total degree: the normal forms of all monomials of the same degree are dealt with using only one call to Gaussian elimination. While this does not require an assumption on the initial ideal, it is unclear to us how to remove the dependency in β in general.

In [FM11, FM17], the authors give faster, probabilistic algorithms for the change of order by means of sparse linear algebra. They do not consider the computation of the multiplication matrices, which are assumed to be known. This approach is not studied in this document, since we do not make any assumption on the multiplication matrices and thus we represent them as dense matrices. For the sake of comparison, we still summarize this approach below; we note however that the cost bounds related to it or the assumptions that one should require to ensure correctness are not clear to us for the moment.

Noticing that the multiplication matrices arising in actual computations are often sparse, [FM11, FM17] tackle Problem 2 from a point of view similar to the Wiedemann algorithm. Choosing some linear functionals to evaluate the monomials in $\mathbb{K}[\boldsymbol{X}]/\mathcal{I}$ allows one to build a multi-dimensional recurrent sequence which admits $\mathcal{I}$ as its ideal of relations (this is only true for some type of ideals $\mathcal{I}$). In terms of the multi-Krylov matrices we are considering in Chapter 4 concerning Problem 2, this is similar to introducing an additional projection on the right of the multiplication matrices to take advantage of the sparsity by using a black-box point of view. Then, recovering a $\prec_{\text{lex}}$-Gröbner basis of this ideal of relations can be done via the Berlekamp-Massey-Sakata algorithm [Sak90], or the recent improvements in [BBF15, BBF17, BF16].

2.3 Multiplication matrix in nilpotent Jordan form

We now consider the specific case of Problem 4 for a multiplication matrix $\mathbf{M} = \mathbf{Z}$ which is block diagonal with nilpotent Jordan blocks, that is, $\mathbf{Z} = \operatorname{diag}(\mathbf{Z}_1, \ldots, \mathbf{Z}_n) \in \mathbb{K}^{D\times D}$ with

$$\mathbf{Z}_i = \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & 0 & 1 \\ & & & 0 \end{bmatrix} \in \mathbb{K}^{D_i \times D_i}$$

for $1 \leqslant i \leqslant n$, and $D = D_1 + \cdots + D_n$. We first give an explicit link between this situation and a generalized form of Hermite-Padé approximation.

2.3.1 Link with minimal approximant bases

We are going to show that finding relation bases for such multiplication matrices corresponds to computing shifted minimal approximant bases, which are also known as σ-bases or order bases [VBB91, BL94, GJV03, ZL12], and which we specify now in Problem 7.

Hereafter, an element of the module $\operatorname{Syz}_{\mathbf{D}}(\mathbf{F})$ in Problem 7 is called an *approximant of* $\operatorname{Syz}_{\mathbf{D}}(\mathbf{F})$, and a basis of this module is called an *approximant basis of* $\operatorname{Syz}_{\mathbf{D}}(\mathbf{F})$.

We now detail the link between Problem 7 and Problem 4 for a multiplication matrix $\mathbf{Z} = \operatorname{diag}(\mathbf{Z}_1, \ldots, \mathbf{Z}_n)$ as above. We first note that this particular case of Problem 4 is of special interest: it is at the core of almost all fast algorithms for operations on polynomial matrices. Furthermore, the two other particular cases of Problem 4 that we study in this thesis, and which are presented in Sections 2.4 and 2.5, are two different generalizations of

Problem 7 – *Minimal approximant basis*

Input:

- positive integers $\mathbf{D} = (D_1, \ldots, D_n) \in \mathbb{Z}^n_{>0}$,
- matrix $\mathbf{F}$ in $\mathbb{K}[X]^{m\times n}$ with $\operatorname{cdeg}(\mathbf{F}) < \mathbf{D}$,
- shift $\mathbf{s} \in \mathbb{Z}^m$.

Output:

- a matrix $\mathbf{P}$ in $\mathbb{K}[X]^{m\times m}$ such that
  - the rows of $\mathbf{P}$ form a basis of the $\mathbb{K}[X]$-module

$$\operatorname{Syz}_{\mathbf{D}}(\mathbf{F}) = \{\mathbf{p} \in \mathbb{K}[X]^{1\times m} \mid \mathbf{p}\mathbf{F}_{*,j} = 0 \bmod X^{D_j} \text{ for } 1 \leqslant j \leqslant n\},$$

  - $\mathbf{P}$ is $\mathbf{s}$-reduced.

Problem 7, where each nilpotent Jordan block is replaced either with an arbitrary Jordan matrix or with a companion matrix.

Since $D = D_1 + \cdots + D_n$, one may identify $\mathcal{M} = \mathbb{K}^D$ with the product of residue class rings

$$\mathcal{F} = \mathbb{K}[X]/(X^{D_1}) \times \cdots \times \mathbb{K}[X]/(X^{D_n}),$$

by mapping a vector $\mathbf{f} = (f_1, \ldots, f_n) \in \mathcal{F}$ to the vector $\mathbf{e} \in \mathcal{M}$ made from the concatenation of the coefficient vectors of $f_1, \ldots, f_n$. Over $\mathcal{F}$, the $\mathbb{K}[X]$-module structure on $\mathcal{M}$ given by $p \cdot \mathbf{e} = \mathbf{e}\, p(\mathbf{Z})$ becomes

$$p \cdot \mathbf{f} = (pf_1 \bmod X^{D_1},\ \ldots,\ pf_n \bmod X^{D_n}).$$

More generally, let $\mathbf{p} \in \mathbb{K}[X]^{1\times m}$ be a polynomial vector and let $\mathbf{E} \in \mathbb{K}^{m\times D}$, whose rows are denoted by $\mathbf{e}_1, \ldots, \mathbf{e}_m \in \mathcal{M}$. By the aforementioned correspondence between $\mathcal{M}$ and $\mathcal{F}$, we associate with $\mathbf{E}$ the matrix $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ whose rows are $\mathbf{f}_1, \ldots, \mathbf{f}_m \in \mathcal{F}$. Then the product $\mathbf{p} \cdot \mathbf{E}$ is the truncated polynomial matrix product $\mathbf{p}\mathbf{F}$, where the column j is taken modulo X^{D_j}. As a consequence, the relation $p_1 \cdot \mathbf{e}_1 + \cdots + p_m \cdot \mathbf{e}_m = 0$ precisely means that $\mathbf{p}\mathbf{F}_{*,j} = 0 \bmod X^{D_j}$ for $1 \leqslant j \leqslant n$.

Thus, we have that a row vector in $\mathbb{K}[X]^{1\times m}$ is a relation of $(\mathbf{Z}, \mathbf{E})$ if and only if it is an approximant of $\operatorname{Syz}_{\mathbf{D}}(\mathbf{F})$, hence the link with Problem 4.

2.3.2 Overview of previous work

The parameter $D = D_1 + \cdots + D_n$, which is the dimension of the input module $\mathcal{M}$ as a $\mathbb{K}$-vector space, is central in the cost bounds of algorithms which solve this problem (see Lemma 2.10 and Section 1.2.2).

First algorithms for Problem 7 with a cost quadratic in D were given in [Ser87, Pas87] in the case of Hermite-Padé approximation ($n = 1$), assuming a type of genericity of $\mathbf{F}$ and outputting a single approximant $\mathbf{p} \in \mathbb{K}[X]^{1\times m}$ which satisfies some prescribed degree constraints. For $n = 1$, van Barel and Bultheel have proposed in [VBB91] an algorithm which uses $\mathcal{O}(m^2 D^2)$ operations to compute an $\mathbf{s}$-minimal basis of approximants

for $\mathbf{F}$ at order D, for any $\mathbf{F}$ and $\mathbf{s}$. This iterative solution was then generalized by Beckermann and Labahn [BL94, Algorithm FPHPS] [BL00, Algorithm FFFG] to eventually deal with Problem 4 for any triangular multiplication matrix. We detail Algorithm FFFG in Section 6.4; it computes the $\mathbf{s}$-Popov approximant basis in $\mathcal{O}(mD^2)$ operations (see Algorithm 8 and Proposition 6.5). In this algorithm, computing the $\mathbf{s}$-Popov basis is a key towards a better control of the degrees in the matrices, which explains the improvement from $\mathcal{O}(m^2D^2)$ to $\mathcal{O}(mD^2)$.

For $D_1 = \cdots = D_n = D/n$, in [BL94] the authors also propose a divide-and-conquer algorithm using $\tilde{\mathcal{O}}(m^\omega D)$ operations in $\mathbb{K}$; the base case of the recursion deals with a scalar column vector using Gaussian elimination. Then, Giorgi et al. [GJV03] followed a similar divide-and-conquer approach, however introducing a base case which has matrix dimensions $m \times n$ and solving it efficiently via fast Gaussian elimination. This leads to an algorithm with cost bound $\mathcal{O}(m^\omega \mathsf{M}(D/n)\log(D/n))$, which reaches our target cost bound $\mathcal{O}(m^{\omega-1}D)$ when $n \in \Theta(m)$.

On the other hand, in the case of Hermite-Padé approximation ($n = 1 \ll m$) it was noticed by Lecerf [Lec01] that the cost bound $\tilde{\mathcal{O}}(m^\omega D)$ is pessimistic, at least when some type of genericity is assumed concerning the input $\mathbf{F}$, and that in this case there is hope to achieve $\tilde{\mathcal{O}}(m^{\omega-1}D)$. This cost bound was then obtained by Storjohann [Sto06], for computing the small degree rows of an $\mathbf{s}$-minimal approximant basis. This relies on the algorithm of [GJV03] via reducing the case of small n and order $D_1 = \cdots = D_n = D/n$ to a situation with larger column dimension $n' \approx m$ and smaller order $D' \approx D/m$.

For efficiency, this reduction requires us to make either of the following assumptions on the shift: $\mathcal{H}_{\mathbf{s},1} : |\mathbf{s} - \min(\mathbf{s})| \in \mathcal{O}(D)$ and $\mathcal{H}_{\mathbf{s},2} : |\max(\mathbf{s}) - \mathbf{s}| \in \mathcal{O}(D)$. As explained in Section 1.2.2, these assumptions imply in particular that any $\mathbf{s}$-minimal basis has size in $\mathcal{O}(mD)$. Then, building upon the reduction technique of [Sto06], Zhou and Labahn designed an algorithm to efficiently compute a full $\mathbf{s}$-minimal basis [Zho12, ZL12].

Proposition 2.16 ([ZL12, Theorems 5.3 and 6.14])**.** *Let $d \in \mathbb{Z}_{>0}$, $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ be of degree less than d, and $\mathbf{s} \in \mathbb{Z}^m$. Let $\mathbf{D} = (d, \ldots, d) \in \mathbb{Z}^n_{>0}$ and $D = nd$. Assuming that $n \leqslant m$, $m \leqslant nd$, and that $\mathcal{H}_{\mathbf{s},1}$ or $\mathcal{H}_{\mathbf{s},2}$ holds, an $\mathbf{s}$-minimal approximant basis of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$ can be computed deterministically using*

$$\mathcal{O}(m^\omega \mathsf{M}(nd/m)\log(d)) = \mathcal{O}(m^\omega \mathsf{M}(D/m)\log(D/n))$$

operations in $\mathbb{K}$. In this cost bound, it is assumed that $\mathsf{M}(\delta) \in \mathcal{O}(\delta^{\omega-1})$.

In the next section we present the minimal approximant basis algorithm proposed in this thesis, which improves upon this result in several ways. Within the same cost bound $\tilde{\mathcal{O}}(m^{\omega-1}D)$, it returns the unique approximant basis in $\mathbf{s}$-Popov form and covers the general case of Problem 7, that is, it works for arbitrary orders $\mathbf{D}$ and shift $\mathbf{s}$.

2.3.3 Computing shifted Popov approximant bases

The next result focuses on the usual case $m \in \mathcal{O}(D)$ and follows from Proposition 7.10 and the corresponding Algorithm 14. We recall the notation $D = D_1 + \cdots + D_n$.

Theorem 2.17. *For $m \in \mathcal{O}(D)$, there is a deterministic algorithm which solves Problem 7 using*

$$\mathcal{O}(m^\omega \mathsf{M}(D/m)\log(D/m)^2 + m^{\omega-1} D \log(m))$$

operations in $\mathbb{K}$ and returns the basis in $\mathbf{s}$-Popov form.

The second term in this cost bound comes from the D/m leaves of the recursion. Each of them uses $m^\omega \log(m)$ operations, by relying on linear algebra (Section 2.2). This term is here to account for the logarithmic factor $\log(m)$ which arises in the cost bound when D is close to m, such as when $D \in \Theta(m)$ or $D \in \Theta(m \log(m))$. As soon as D is significantly larger than m, this term is negligible in front of the first one.

As said above, when $D \in \mathcal{O}(m)$, one may rely on the fast algorithm based on linear algebra which was presented in Section 2.2. Since in the context here the degree of the minimal polynomial of the multiplication matrix $\mathbf{Z} = \operatorname{diag}(\mathbf{Z}_1, \ldots, \mathbf{Z}_n)$ is $\max(\mathbf{D})$, the $\mathbf{s}$-Popov approximant basis can be computed in $\mathcal{O}(mD^{\omega-1} + D^\omega \log(\max(\mathbf{D})))$ operations in $\mathbb{K}$, according to Theorem 2.12.

We note that the case $n \leqslant D \leqslant m$ was studied in [Zho12, Section 3.6], where Zhou gives an algorithm whose announced cost bound is $\tilde{\mathcal{O}}(m^{\omega-1}D)$; however a more precise analysis shows that the cost is $\mathcal{O}(D^{\omega-1}m + D^\omega \log(D/n))$ as in the previous paragraph. However, this algorithm only works in the case of the uniform shift and identical orders $D_1 = \cdots = D_n = D/n$.

Coming back to $m \in \mathcal{O}(D)$, we have seen in Proposition 2.16 that slightly faster algorithms exist in the literature, with one less logarithm factor. Still, these algorithms assume identical orders $\mathbf{D}$ and a shift $\mathbf{s}$ such that $\mathcal{H}_{\mathbf{s},1}$ or $\mathcal{H}_{\mathbf{s},2}$; besides, for most inputs, they do not return a basis in shifted Popov form. Yet, having the shifted Popov basis is convenient for many reasons (see Sections 1.1.3 and 1.2.2); for example, it has been exploited in [RS16, Theorem 12]. Furthermore, there are known situations where one is interested in dealing with arbitrary orders $\mathbf{D} = (D_1, \ldots, D_n)$, for example in the computation of kernel bases with information on the output degrees, as in [GS11, Section 3] and Section 8.1. Finally, the assumptions $\mathcal{H}_{\mathbf{s},1}$ and $\mathcal{H}_{\mathbf{s},2}$ on the shift are quite restrictive; notably, removing this assumption allows us to cover the computation of approximant bases in Hermite form.

Overview of our algorithm. In short, our fast shifted Popov approximant basis algorithm combines our strategy to find and use the $\mathbf{s}$-minimal degree based on the divide-and-conquer approach of [BL94] (see Section 1.2.1) with the partial linearization of [Sto06] to reduce to a case where the degrees are well controlled. We now give more details.

In our fast algorithm for Problem 7, we use a slightly modified version of the algorithm [GJV03, PM-Basis] which we present in details in Section 7.1. This is efficient in the case of identical orders and $m \in \mathcal{O}(n)$, and our modified version ensures that the computed basis is in shifted *ordered weak Popov* form, instead of being simply $\mathbf{s}$-reduced. The advantage of an ordered weak Popov form is that it allows us to directly read the $\mathbf{s}$-pivot degrees. In Section 7.2, we use this modified [GJV03, PM-Basis] to design an algorithm which, from an arbitrary instance of Problem 7 with $n \geqslant m$, computes an approximant basis for sufficiently many of the n equations so that it essentially remains to solve an instance of Problem 7 with $n < m$.

Then, in Section 7.3, we consider solving Problem 7, and more precisely computing the $\mathbf{s}$-Popov approximant basis, when the $\mathbf{s}$-minimal degree $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$ of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$ is known a priori. We recall from Definition 1.22 that $\boldsymbol{\delta}$ corresponds to the $\mathbf{s}$-pivot degree of the $\mathbf{s}$-Popov approximant basis of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$. As explained in Section 1.2.1, knowing $\boldsymbol{\delta}$ allows us to rather consider the shift $-\boldsymbol{\delta}$, for which $-\boldsymbol{\delta}$-minimal approximant bases have good properties. In particular, they have column degree $\boldsymbol{\delta}$ and therefore size in $\mathcal{O}(mD)$, and the sought $\mathbf{s}$-Popov basis can be easily retrieved from any $-\boldsymbol{\delta}$-minimal basis.

Thus, we focus on computing a $-\boldsymbol{\delta}$-minimal approximant basis; in this case, thanks to the knowledge of the column degree $\boldsymbol{\delta}$ of the output, we will be able to efficiently use the partial linearization techniques of [Sto06]. To summarize, knowing $\boldsymbol{\delta}$ essentially leads us to partially linearize the arbitrary instance of Problem 7 into an instance with identical orders and dimensions $n \in \Theta(m)$. In the latter situation, the algorithm of [GJV03] can be used to efficiently compute a shifted minimal approximant basis.

Finally, our main algorithm relies on adapting the divide-and-conquer approach of [BL94, GJV03] to efficiently find the $\mathbf{s}$-minimal degree $\boldsymbol{\delta}$. The base case of the recursion is when $D = D_1 + \cdots + D_n \in \mathcal{O}(m)$: then, we can efficiently compute the $\mathbf{s}$-Popov basis via linear algebra as in Theorem 2.12, and this basis gives us $\boldsymbol{\delta}$. Then, when $D > m$, we split the problem into two subproblems of similar size as in Example 1.27: roughly, we deal with the lower $D/2$ coefficients of the input $\mathbf{F}$ in the first subproblem, and with the higher $D/2$ coefficients in the second. Then, we recursively compute shifted Popov approximant bases $\mathbf{P}^{(1)}$ and $\mathbf{P}^{(2)}$ for these subproblems.

Following [BL94, GJV03], one would then compute and return the product $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$, which is an $\mathbf{s}$-minimal approximant basis. However, in general its degrees may be too large for efficient computation, as we show in Example 7.5. Thus, instead of computing this product, we rely on the item (iv) of Theorem 1.28 to deduce $\boldsymbol{\delta}$ as being the sum of the diagonal degrees of $\mathbf{P}^{(1)}$ and $\mathbf{P}^{(2)}$. Then, we compute the global $\mathbf{s}$-Popov approximant basis from scratch, using the original instance of Problem 7 along with the additional knowledge of $\boldsymbol{\delta}$.

Remark 2.18. Here is a natural question, which is not tackled in this thesis: for identical orders and a shift $\mathbf{s}$ such that $\mathcal{H}_{\mathbf{s},1}$ or $\mathcal{H}_{\mathbf{s},2}$, how to compute the $\mathbf{s}$-Popov approximant basis with a cost bound similar to that of the algorithms of [ZL12] recalled in Proposition 2.16, that is, with only one logarithm factor?

One approach is to first compute an $\mathbf{s}$-minimal approximant basis $\mathbf{P}$ using the algorithm in [ZL12], and then normalize it into $\mathbf{s}$-Popov form using the algorithm in [SS11]. Yet, this would not lead to the desired complexity as such, since the row degrees in $\mathbf{P}$ may be unbalanced, leading to a worst-case cost bound exceeding the target $\mathcal{O}^{\sim}(m^{\omega-1}D)$. It is not clear whether the algorithm in [SS11] can be extended to take into account the unbalancedness of the row degrees in the input.

Another approach is to rely on the a priori knowledge of the $\mathbf{s}$-minimal degree $\boldsymbol{\delta}$ of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$. If $\boldsymbol{\delta}$ is known, then Algorithm 13 uses $\mathcal{O}(m^{\omega}\mathsf{M}(D/m)\log(D/m))$ operations to compute the $\mathbf{s}$-Popov basis, according to Proposition 7.9. Judging from the material in Chapter 7, it seems reasonable to expect that with a minor modification, the algorithms in [ZL12] would return an approximant basis $\mathbf{P}$ in $\mathbf{s}$-ordered weak Popov form. In this case, we would directly obtain $\boldsymbol{\delta}$ by reading the degrees of the diagonal entries of $\mathbf{P}$. ☕

2.4 Multiplication matrix in Jordan form

In this section, we deal with the case of a multiplication matrix $\mathbf{M} = \mathbf{J}$ in Jordan form. We first detail the correspondence between relations for such a multiplication matrix and some type of interpolants. Then, we present previous work and our results on the computation of these interpolants.

2.4.1 Link with minimal interpolant bases

Here, we show that in the context of a multiplication matrix in Jordan form, the notion of relation is directly related to the notion of M-Padé approximant, sometimes also called interpolants. The problem of computing M-Padé approximants was studied in [Lüb83, Bec90, Bec92, VBB92] and named after the work of Mahler, including in particular [Mah32, Mah53, Mah68]. We also refer the reader to [Coa66, Coa67, Jag64].

Let $\mathbf{J} \in \mathbb{K}^{D\times D}$ be a Jordan matrix with n diagonal Jordan blocks of respective sizes $D_1, \ldots, D_n$ and with respective eigenvalues $x_1, \ldots, x_n$. Following the ideas in Section 2.3.1, one may identify $\mathcal{M} = \mathbb{K}^D$ with

$$\mathcal{F} = \mathbb{K}[X]/(X^{D_1}) \times \cdots \times \mathbb{K}[X]/(X^{D_n}),$$

by mapping a vector $\mathbf{f} = (f_1, \ldots, f_n)$ in $\mathcal{F}$ to the vector $\mathbf{e} \in \mathcal{M}$ made from the concatenation of the coefficient vectors of $f_1, \ldots, f_n$. Then, over $\mathcal{F}$, the $\mathbb{K}[X]$-module structure on $\mathcal{M}$ given by $p \cdot \mathbf{e} = \mathbf{e}\, p(\mathbf{J})$ becomes

$$p \cdot \mathbf{f} = (p(X + x_1)f_1 \bmod X^{D_1}, \ldots, p(X + x_n)f_n \bmod X^{D_n}).$$

Now, if $(\mathbf{e}_1, \ldots, \mathbf{e}_m)$ in $\mathcal{M}^m$ is associated to $(\mathbf{f}_1, \ldots, \mathbf{f}_m)$ in $\mathcal{F}^m$, with $\mathbf{f}_i = (f_{i,1}, \ldots, f_{i,n})$ and $f_{i,j}$ in $\mathbb{K}[X]/(X^{D_j})$ for all i, j, the relation $p_1 \cdot \mathbf{e}_1 + \cdots + p_m \cdot \mathbf{e}_m = 0$ means that for all j in $\{1, \ldots, n\}$, we have

$$p_1(X + x_j)f_{1,j} + \cdots + p_m(X + x_j)f_{m,j} = 0 \bmod X^{D_j};$$

applying a translation by $-x_j$, this is equivalent to

$$p_1 f_{1,j}(X - x_j) + \cdots + p_m f_{m,j}(X - x_j) = 0 \bmod (X - x_j)^{D_j}.$$

Thus, in terms of vector M-Padé approximation as in [VBB92], $(p_1, \ldots, p_m)$ is an interpolant for $(\mathbf{f}_1, \ldots, \mathbf{f}_m)$, $x_1, \ldots, x_n$, and $D_1, \ldots, D_n$.

Then, the problem of computing relation bases for a multiplication matrix in Jordan form can be rewritten as follows.

Let us write $\mathbf{D} = (D_1, \ldots, D_n)$, and let $D = D_1 + \cdots + D_n$. Then, the data of the points $x_1, \ldots, x_n$ and of the integers $\mathbf{D}$ directly define the Jordan multiplication matrix $\mathbf{J} \in \mathbb{K}^{D\times D}$, as explained above. In what follows, the module in Problem 8 is denoted by $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$; an element $\mathbf{p} \in \mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$ is called an *interpolant for* $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$, and a matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ whose rows form a basis of $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$ is called an *interpolant basis of* $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$.

Our main results concerning the case of a Jordan multiplication matrix, stated in Theorems 2.19 and 2.20, can be summarized as follows: there is a deterministic algorithm which solves Problem 8 and returns the $\mathbf{s}$-Popov basis using $\tilde{\mathcal{O}}(m^{\omega-1}D)$ operations in $\mathbb{K}$.

Problem 8 – *Minimal interpolant basis*

Input:

- points $x_1, \ldots, x_n$ in $\mathbb{K}$,
- positive integers $\mathbf{D} \in \mathbb{Z}^n_{>0}$,
- matrix $\mathbf{F}$ in $\mathbb{K}[X]^{m\times n}$ such that $\operatorname{cdeg}(\mathbf{F}) < \mathbf{D}$,
- shift $\mathbf{s} \in \mathbb{Z}^m$.

Output:

- a matrix $\mathbf{P}$ in $\mathbb{K}[X]^{m\times m}$ such that
  - the rows of $\mathbf{P}$ form a basis of the $\mathbb{K}[X]$-module
  $$\{\mathbf{p} \in \mathbb{K}[X]^{1\times m} \mid \mathbf{p}\mathbf{F}_{*,j} = 0 \bmod (X - x_j)^{D_j} \text{ for } 1 \leqslant j \leqslant n\},$$
  - $\mathbf{P}$ is $\mathbf{s}$-reduced.

Previous work on this problem includes [Bec92, VBB92, BL00]. We note that in [Bec92], the input consists of a single column $\mathbf{F}$ in $\mathbb{K}[X]^{m\times 1}$ of degree less than D: to form the input of our problem, we compute $\hat{\mathbf{F}} = [\mathbf{F} \bmod (X - x_1)^{D_1} | \cdots | \mathbf{F} \bmod (X - x_n)^{D_n}]$.

For an arbitrary shift, the algorithms in [Bec92, VBB92] have a cost of $\mathcal{O}(m^2D^2)$ operations and return a shifted minimal interpolant basis. In [BL00, Algorithm FFFG], the degrees are better controlled by ensuring that the bases computed at each iteration of the algorithm are in shifted Popov form; this leads to the cost bound $\mathcal{O}(mD^2)$ to compute the $\mathbf{s}$-Popov interpolant basis. To the best of our knowledge, no algorithm with a cost quasi-linear in D had been given in the literature prior to our work.

As we observe from our review of previous work on interpolant bases and the more specific approximant bases, the design of fast algorithms mainly focused on the latter. Here, relying on the general framework of relation bases developed in [BL00], we manage to give fast algorithms for the general case of interpolant basis computation. The cost bounds for these algorithms match, up to logarithmic factors, those of fast algorithms for approximant bases presented in Section 2.3.

2.4.2 Algorithm for almost uniform shifts

Here, we present our fast algorithm in $\tilde{\mathcal{O}}(m^{\omega-1}(D+\xi))$ for computing $\mathbf{s}$-minimal relation bases with a multiplication matrix in Jordan form, where $\xi = |\mathbf{s} - \min(\mathbf{s})|$. We recall from Section 1.2.2 and Lemma 2.10 that this quantity ξ provides an a priori upper bound on the sum of the row degrees of these $\mathbf{s}$-minimal bases.

Result. If $\mathbf{J} \in \mathbb{K}^{D\times D}$ is a Jordan matrix with n diagonal blocks of respective sizes $D_1, \ldots, D_n$ and with respective eigenvalues $x_1, \ldots, x_n$, we will write it in a compact manner by specifying only those sizes and eigenvalues. More precisely, we will assume that $\mathbf{J}$ is given to us as the form

$$\mathbf{J} = ((x_1, D_{1,1}), \ldots, (x_1, D_{1,r_1}), \ldots, (x_t, D_{t,1}), \ldots, (x_t, D_{t,r_t})), \tag{2.3}$$

for some pairwise distinct $x_1, \ldots, x_t$, with $r_1 \geqslant \cdots \geqslant r_t$ and $D_{i,1} \geqslant \cdots \geqslant D_{i,r_i}$ for all i; we will say that this representation is *standard*. If $\mathbf{J}$ is given as an arbitrary list $((x_1, D_1), \ldots, (x_n, D_n))$, we can reorder it (and from that, permute the columns of $\mathbf{F}$ accordingly) to bring it to the above form in time $\mathcal{O}(\mathsf{M}(D)\log(D)^3)$ using the algorithm in [BJS08, Proposition 12]; if $\mathbb{K}$ is equipped with an order, and if we assume that comparisons take unit time, it is of course enough to sort the x_i's.

Then, our first algorithm for the case of a multiplication matrix in Jordan form achieves the following result, which focuses on the usual case $m \in \mathcal{O}(D)$. Concerning the case $D \in \mathcal{O}(m)$, we rely on the solution based on linear algebra presented in Section 2.2 and the corresponding cost bound can be found in Theorem 2.12. Besides, the reader interested in the logarithmic factors may refer to the more precise cost bound in Proposition 13.3.

Theorem 2.19. *Let $\mathbf{J} \in \mathbb{K}^{D \times D}$ be a Jordan matrix given by a standard representation. Then, there is a deterministic algorithm which solves Problem 4 with $\mathbf{M} = \mathbf{J}$ using*

$$\mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(D)\log(D/m) + m^{\omega-1}\mathsf{M}(\xi)\log(\xi/m))$$

operations in $\mathbb{K}$, where $\xi = |\mathbf{s} - \min(\mathbf{s})|$.

We remark that under the assumption $\mathcal{H}_{\mathbf{s},1} : \xi \in \mathcal{O}(D)$ which states that the shift is almost uniform around its minimal value, this cost bound becomes $\mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(D))$, which is in $\mathcal{O}\tilde{}(m^{\omega-1}D)$.

In general, masking logarithmic factors, this cost bound is $\mathcal{O}\tilde{}(m^{\omega-1}(D+\xi))$. On the other hand, following the discussion in Section 1.2.2, the bound of Lemma 2.10 on the degree of the determinant of $\mathbf{s}$-minimal interpolant bases implies that the sum of row degrees of the output basis is in $\mathcal{O}(D+\xi)$, and thus its size is in $\mathcal{O}(m(D+\xi))$.

To give a worst-case cost bound independently of the shift, we recall from Section 1.2.2 that one may assume without loss of generality that $\min(\mathbf{s}) = 0$, $\max(\mathbf{s}) \in \mathcal{O}(mD)$, and $|\mathbf{s}| \in \mathcal{O}(m^2 D)$. We also introduced there the shift $\mathbf{h} = (0, D, 2D, \ldots, (m-1)D)$, for which these bounds are reached and the $\mathbf{h}$-Popov interpolant basis is actually the interpolant basis in Hermite form. This is thus a worst-case situation for the cost bound $\mathcal{O}\tilde{}(m^{\omega-1}(D+\xi))$, which with $\xi = |\mathbf{h}|$ becomes $\mathcal{O}\tilde{}(m^{\omega+1}D)$.

As detailed in Section 2.3.2, the cost bound $\mathcal{O}\tilde{}(m^{\omega-1}D)$ has been achieved in the specific case of a *nilpotent* Jordan matrix $\mathbf{J}$, with identical block sizes and under assumptions on $\mathbf{s}$ in [GJV03, ZL12], and in the general case here (Theorem 2.17). However, we are not aware of a previous cost bound that would be similar to the result above, for example that would be quasi-linear in D, for the question of Problem 4 with an *arbitrary* Jordan matrix.

An interesting case which involves a non-nilpotent Jordan matrix can be found in applications of Problem 4 to multivariate interpolation and list-decoding algorithms, as detailed in Section 3.1.

In fact, it was left as an open problem in [ZL12, Section 7] to obtain algorithms with cost bound $\mathcal{O}\tilde{}(m^{\omega-1}D)$ for arbitrary shifts $\mathbf{s}$ and in such a general interpolation context. With the result above and the one to follow in Section 2.4.3 about arbitrary shifts, we solve this open problem.

Overview of the algorithm. Our divide-and-conquer algorithm is given in Section 13.2. The idea is to use a Knuth-Schönhage-Moenck half-gcd approach [Knu70, Sch71, Moe73], previously carried over to the particular case of minimal approximant bases in [BL94, GJV03]. This approach consists in reducing a problem in size D to a first subproblem in size $D/2$, the computation of the so-called *residual*, a second subproblem in size $D/2$, and finally a recombination of the results of both subproblems via polynomial matrix multiplication. The shift to be used in the second recursive call is essentially the $\mathbf{s}$-row degree of the outcome of the first recursive call.

The main difficulty is to control the degrees in the interpolant bases that are obtained recursively. We have at hand the bound $\mathcal{O}(D+\xi)$ on the sum of row degrees: it depends on the input shift. Yet, in our algorithm, we cannot make any assumption on the shifts that will appear in recursive calls, since they are related to the degrees of the entries of the previously computed bases. Hence, even in the case of a uniform input shift for which the output basis is of size $\mathcal{O}(mD)$, there may be recursive calls with essentially arbitrary shifts, which may output bases that have a size too large with respect to our target cost.

Our workaround is to perform all recursive calls with the uniform shift $\mathbf{s} = \mathbf{0}$, and resort to a change of shift that will be studied in Section 14.1. This strategy is an alternative to the partial linearization approach that was used in [Sto06, ZL12] in the case of approximant bases, whose generalization to interpolant bases is not clear to us. To clarify the word "alternative", we note that our change of shift uses a kernel basis algorithm which itself relies on approximant basis computations; yet, with the dimensions involved here, this approximation problem is solved efficiently without resorting to [ZL12].

Another difficulty which was not solved prior to this work is to deal with instances where D is small. Our bound on the size of the output in Section 1.2.2 states that when $D \leqslant m$ and the shift is uniform, the average degree of the entries of a minimal interpolant basis is at most 1. Thus, in this case our focus is not anymore on using fast polynomial arithmetic but rather on exploiting efficient linear algebra over $\mathbb{K}$: the divide-and-conquer process stops when reaching $D \leqslant m$, and invokes instead the algorithm based on linear algebra discussed above in Section 2.2 and detailed in Chapter 4.

The last ingredient is the fast computation of the residual, that is, a matrix in $\mathbb{K}^{m\times D/2}$ for restarting the process after having found a basis $\mathbf{P}^{(1)}$ for the first subproblem of size $D/2$. This boils down to computing $\mathbf{P}^{(1)} \cdot \mathbf{F}$ and discarding the first $D/2$ columns, which are known to be zero. In Section 14.2, we design a general procedure for computing this kind of product, using Hermite interpolation and evaluation to reduce it to multiplying polynomial matrices.

Concerning the multiplication of the bases obtained recursively, to handle the fact that they may have unbalanced row degrees, we use the approach in [ZLS12, Section 3.6]; we give a detailed algorithm and cost analysis in Section 12.2.

2.4.3 Computing shifted Popov interpolant bases

Now, we present our second algorithm for the case of a multiplication matrix in Jordan form. It uses the one presented in the previous section as a building block, and offers two key new features: it supports *arbitrary* shifts with a cost $\tilde{\mathcal{O}}(m^{\omega-1}D)$, and it computes the basis *in* $\mathbf{s}$-*Popov form*.

Result. Again, for small values of D, namely $D \in \mathcal{O}(m)$, these features are already obtained by the approach based on linear algebra presented in Section 2.2, with a cost bound that is satisfactory. Hence, here, we focus on the case $m \in \mathcal{O}(D)$.

Theorem 2.20. *Let $\mathbf{J} \in \mathbb{K}^{D \times D}$ be a Jordan matrix given by a standard representation. Then, there is a deterministic algorithm which solves Problem 4 with $\mathbf{M} = \mathbf{J}$ using*

$$\mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(D)\log(D/m)^2)$$

operations in $\mathbb{K}$ and returns the basis in $\mathbf{s}$-Popov form.

Since the worst-case of the cost bound of Theorem 2.19 for the shift-dependent algorithm was $\tilde{\mathcal{O}}(m^{\omega+1}D)$, in general the result here offers a speedup factor of up to m^2. As discussed above in Sections 2.3 and 2.4.1, to the best of our knowledge, no algorithm for this problem with cost $\tilde{\mathcal{O}}(m^{\omega-1}D)$ was known previously for *arbitrary* shifts, even in the specific case of approximant bases.

Overview of our approach. Several previous algorithms for approximant basis computation, such as those in [BL94, GJV03], follow a divide-and-conquer scheme inspired by the Knuth-Schönhage-Moenck algorithm [Knu70, Sch71, Moe73]. The result here builds on the algorithm presented in Section 2.4.2, where we extended this recursive approach to more general interpolation problems involving a general Jordan matrix $\mathbf{J}$, yet without handling arbitrary shifts with a satisfactory complexity.

An immediate challenge is that for an arbitrary shift $\mathbf{s}$, the sum of the row degrees an $\mathbf{s}$-minimal interpolant basis is in $\mathcal{O}(D+\xi)$ where $\xi = |\mathbf{s} - \min(\mathbf{s})|$, and thus its size in $\mathcal{O}(m(D+\xi))$ may be beyond our target cost when ξ is large. However, a first remark is that this bound $\mathcal{O}(D+\xi)$ is pessimistic if we restrict our view to the bases actually computed by the above-mentioned divide-and-conquer approach, which we describe in more precise terms now.

Let $\mathbf{F}$, $\mathbf{J}$, and $\mathbf{s}$ be our input, and write $\mathbf{J}^{(1)}$ and $\mathbf{J}^{(2)}$ for the $D/2 \times D/2$ leading and trailing principal submatrices of $\mathbf{J}$. First, compute an $\mathbf{s}$-minimal interpolant basis $\mathbf{P}^{(1)}$ for $\mathbf{J}^{(1)}$ and the first $D/2$ columns of $\mathbf{F}$; then, compute the last $D/2$ columns $\mathbf{F}^{(2)}$ of the *residual* $\mathbf{P}^{(1)} \cdot \mathbf{F}$; then, compute a $\mathbf{t}$-minimal interpolant basis $\mathbf{P}^{(2)}$ for $(\mathbf{F}^{(2)}, \mathbf{J}^{(2)})$ with $\mathbf{t} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}^{(1)})$; finally, return the matrix product $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$.

In the case of Hermite-Padé approximation, this is exactly the divide-and-conquer algorithm in [BL94]; our algorithm presented in Section 2.4.2 introduces an additional change of shift technique to obtain efficiency for almost uniform shifts. An $\mathbf{s}$-minimal basis computed by this method has degree at most D and thus size in $\mathcal{O}(m^2D)$, which is less than $\mathcal{O}(m(D+\xi))$ when ξ is large. Using our fast algorithm for residual computation, this approach allows us to solve Problem 4 using $\tilde{\mathcal{O}}(m^{\omega}D)$ operations for an arbitrary shift (see Section 13.1), which improves upon our algorithm of Section 2.4.2 when ξ is large.

However, this approach has intrinsic limitations, since there are instances of Problem 4 for which the size of the basis it outputs reaches $\Theta(m^2D)$: this is beyond our target cost $\tilde{\mathcal{O}}(m^{\omega-1}D)$. We detail this in Example 7.5, with an instance of approximant basis computation such that $\xi = \Theta(mD)$.

In Section 2.4.2, focusing on the case where ξ is small compared to D, and preserving such a property in recursive calls via changes of shifts, we obtained the cost bound $\tilde{\mathcal{O}}(m^{\omega-1}(D+\xi))$. For ξ in $\mathcal{O}(D)$, this matches our target cost. The fundamental reason for this kind of improvement over $\tilde{\mathcal{O}}(m^{\omega}D)$, already seen with [ZL12], is that one controls the average row degree of the bases $\mathbf{P}^{(2)}$ and $\mathbf{P}^{(1)}$ and of their product $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$.

From the discussion in Section 1.2.2, we know that we have no such control of the average row degree for an arbitrary shift $\mathbf{s}$. Still, the bound Lemma 2.10 implies that the $\mathbf{s}$-Popov interpolant basis $\mathbf{P}$ for $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$ has average *column* degree at most D/m, and thus its size is in $\mathcal{O}(mD)$. Therefore, our answer to the difficulty that bases may have size in $\Theta(m^2D)$ is to compute a basis in $\mathbf{s}$-Popov form. This clarifies our claim in Section 1.1 that our motivation for computing a basis in shifted Popov form is not only because of the obvious interest of having a canonical form, but also because it helps us to better control the degrees in the manipulated matrices and thus to give faster algorithms.

Then, similarly to our Popov approximant basis algorithm in Section 2.3, we will adapt the above divide-and-conquer strategy to find the $\mathbf{s}$-minimal degree $\boldsymbol{\delta}$ of $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$; this is detailed in Section 13.3. With this additional knowledge, we will rely on the case of almost uniform shifts to efficiently compute the sought $\mathbf{s}$-Popov basis. This follows ideas that we exposed in Section 1.2.1 and which we recall now.

Suppose that we have computed recursively the bases $\mathbf{P}^{(2)}$ and $\mathbf{P}^{(1)}$ in $\mathbf{s}$- and $\mathbf{t}$-Popov form; we want to output the $\mathbf{s}$-Popov form $\mathbf{P}$ of $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$. In general, this product is not normalized and may have size $\Theta(m^2D)$: its computation is beyond our target cost. Thus, one main idea is that we will not rely on polynomial matrix multiplication to combine the bases obtained recursively. Instead, we use $\mathbf{P}^{(1)}$ and $\mathbf{P}^{(2)}$ to find $\boldsymbol{\delta}$ as the sum of their diagonal degrees (see the item (iv) of Theorem 1.28), and then we obtain $\mathbf{P}$ from a minimal interpolant basis computation for the shift $-\boldsymbol{\delta}$.

The latter situation, namely the problem of computing the $\mathbf{s}$-Popov interpolant basis $\mathbf{P}$ for $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$ having the $\mathbf{s}$-minimal degree $\boldsymbol{\delta}$ as an additional input, is studied in Section 14.3. According to Lemma 1.26, this reduces to the computation of a $-\boldsymbol{\delta}$-minimal interpolant basis $\mathbf{R}$. The properties of this shift $-\boldsymbol{\delta}$ allow us first to compute $\mathbf{R}$ in $\tilde{\mathcal{O}}(m^{\omega-1}D)$ operations using the partial linearization framework from [Sto06, Section 3] and the minimal interpolant basis algorithm from Section 2.4.2, and second to easily retrieve $\mathbf{P}$ from $\mathbf{R}$.

We stop the recursion as soon as $D \leqslant m$, in which case we do not need $\boldsymbol{\delta}$ to achieve efficiency: the algorithm presented in Section 2.2 computes the $\mathbf{s}$-Popov interpolant basis in $\tilde{\mathcal{O}}(D^{\omega-1}m)$ operations for any $\mathbf{s}$ according to Theorem 2.12.

2.5 Companion-block diagonal multiplication matrix

We now consider the computation of relation bases with a multiplication matrix $\mathbf{M} = \mathbf{C}$ which is block diagonal with companion blocks. We first establish a correspondence between this problem and that of finding solutions to systems of linear modular equations; as a consequence, approximant bases and more generally interpolant bases can be seen as a particular case of this situation. Then, we discuss previous work and we give our main result and an overview of our algorithm. Finally, we discuss another algorithm based

on structured $\mathbb{K}$-linear algebra, which obtains slightly better efficiency by focusing on returning only one solution which satisfies degree constraints.

2.5.1 Link with systems of linear modular equations

Let us first define precisely what we mean by *systems of linear modular equations*. In what follows, $\mathbb{K}[X]_{\neq 0}$ stands for the set of nonzero polynomials in $\mathbb{K}[X]$, and we denote by $\mathfrak{M} = (\mathfrak{m}_1, \ldots, \mathfrak{m}_n) \in \mathbb{K}[X]^n_{\neq 0}$ a tuple of polynomials that will play the role of moduli. Then, for $\mathbf{A}, \mathbf{B} \in \mathbb{K}[X]^{m \times n}$ we write $\mathbf{A} = \mathbf{B} \bmod \mathfrak{M}$ if there exists $\mathbf{Q} \in \mathbb{K}[X]^{m \times n}$ such that $\mathbf{A} = \mathbf{B} + \mathbf{Q}\mathrm{diag}(\mathfrak{M})$. This means that the column j of $\mathbf{A}$ is equal to the column j of $\mathbf{B}$ up to multiples of $\mathfrak{m}_j$. Then, given a matrix $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ which specifies the equations, we consider the module

$$\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F}) = \{\mathbf{p} \in \mathbb{K}[X]^{1 \times m} \mid \mathbf{pF} = 0 \bmod \mathfrak{M}\}.$$

An element of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ is called a *solution of* $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$).

Example 2.21. Consider the case where $\mathfrak{m}_j = X^{D/n}$ for $1 \leqslant j \leqslant n$. Then for any $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$, $\mathbf{p}$ is a solution of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ if and only if $\mathbf{pF} = 0 \bmod X^{D/n}$.

More generally, when the moduli in $\mathfrak{M}$ are powers of the variable X, the notion of solution of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ coincides with the notion of approximant of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$, where the orders $\mathbf{D}$ are given by these powers. ✍

Since the set $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ is a $\mathbb{K}[X]$-submodule of $\mathbb{K}[X]^{1 \times m}$ which contains the submodule $\mathrm{lcm}(\mathfrak{m}_1, \ldots, \mathfrak{m}_n)\mathbb{K}[X]^{1 \times m}$, $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ is free of rank m according to Lemma 1.1. Then, we may represent any basis of this module as the rows of a matrix $\mathbf{P} \in \mathbb{K}[X]^{m \times m}$, called a *solution basis of* $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$. Such a basis $\mathbf{P}$ which is $\mathbf{s}$-reduced is said to be an $\mathbf{s}$-*minimal* solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$, and if it is furthermore in $\mathbf{s}$-Popov form, then $\mathbf{P}$ is called the $\mathbf{s}$-*Popov* solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$.

The problem of computing minimal bases of solutions to such systems of linear modular equations is stated in Problem 9. Our interest in computing a whole basis in shifted reduced form, and not simply a single solution with degree constraints, lies for example in the application to the computation of shifted normal forms of polynomial matrices; this situation is described in detail in Section 3.2.2. Still, in some contexts we are satisfied with a single small degree solution; we discuss this point in Section 2.5.3.

Problem 9 – *Minimal solution basis*

Input:

- polynomials $\mathfrak{M} = (\mathfrak{m}_1, \ldots, \mathfrak{m}_n) \in \mathbb{K}[X]^n_{\neq 0}$,
- a matrix $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ such that $\deg(\mathbf{F}_{*,j}) < \deg(\mathfrak{m}_j)$,
- a shift $\mathbf{s} \in \mathbb{Z}^m$.

Output:

- an $\mathbf{s}$-minimal solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$.

Well-known specific cases of Problem 9 are Hermite-Padé approximation with a single equation modulo some power of X. More generally, for $\mathbf{D} = (D_1, \ldots, D_n) \in \mathbb{Z}_{>0}$, an

$\mathbf{s}$-minimal approximant basis of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$ as in Problem 7 is an $\mathbf{s}$-minimal solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ with $\mathfrak{M} = (X^{D_1}, \ldots, X^{D_n})$. Even more generally, computing interpolant bases as in Problem 8 also amounts to solving Problem 9, in the specific case of moduli that are products of known linear factors.

Now, let $\mathbf{C} \in \mathbb{K}^{D \times D}$ be a companion-block diagonal matrix with n diagonal blocks. Explicitly,

$$\mathbf{C} = \mathrm{diag}(\mathbf{C}_1, \ldots, \mathbf{C}_n) \text{ with } \mathbf{C}_j = \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & 0 & 1 \\ -c_j^{(0)} & -c_j^{(1)} & \cdots & -c_j^{(D_j-1)} \end{bmatrix} \in \mathbb{K}^{D_j \times D_j}, \tag{2.4}$$

for $1 \leqslant j \leqslant n$. We have in particular $D = D_1 + \cdots + D_n$.

Then, defining the monic polynomial $\mathfrak{m}_j = c_j^{(0)} + c_j^{(1)} X + \cdots + c_j^{(D_j-1)} X^{D_j-1} + X^{D_j}$ for $1 \leqslant j \leqslant n$, one may identify $\mathcal{M} = \mathbb{K}^D$ with the product of residue class rings

$$\mathcal{F} = \mathbb{K}[X]/(\mathfrak{m}_1) \times \cdots \times \mathbb{K}[X]/(\mathfrak{m}_n),$$

by mapping a vector $\mathbf{f} = (f_1, \ldots, f_n)$ in $\mathcal{F}$ to the vector $\mathbf{e} \in \mathcal{M}$ made from the concatenation of the coefficient vectors of $f_1, \ldots, f_n$. Then, over $\mathcal{F}$, the $\mathbb{K}[X]$-module structure on $\mathcal{M}$ given by $p \cdot \mathbf{e} = \mathbf{e}\, p(\mathbf{C})$ becomes

$$p \cdot \mathbf{f} = (pf_1 \bmod \mathfrak{m}_1, \ \ldots, \ pf_n \bmod \mathfrak{m}_n).$$

If vectors $\mathbf{e}_1, \ldots, \mathbf{e}_m$ of $\mathcal{M}$, which form the rows of a matrix $\mathbf{E} \in \mathbb{K}^{m \times D}$, are associated with polynomials $\mathbf{f}_1, \ldots, \mathbf{f}_m$ in $\mathcal{F}$, which form the rows of a matrix $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ with $\mathrm{cdeg}(\mathbf{F}_{*,j}) < \deg(\mathfrak{m}_j)$, then the relation $p_1 \cdot \mathbf{e}_1 + \cdots + p_m \cdot \mathbf{e}_m = 0$ precisely means that

$$\mathbf{p}\mathbf{F}_{*,j} = 0 \bmod \mathfrak{m}_j \text{ for } 1 \leqslant j \leqslant n.$$

Thus, defining $\mathfrak{M} = (\mathfrak{m}_1, \ldots, \mathfrak{m}_n)$, a row vector $\mathbf{p} \in \mathbb{K}[X]^{1 \times m}$ is a relation of $\mathrm{Syz}_{\mathbf{C}}(\mathbf{E})$ if and only if $\mathbf{p}$ is a solution of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$.

Seen the other way, if $\mathfrak{M}$ and $\mathbf{F}$ are as in the input of Problem 9, we let $D_j = \deg(\mathfrak{m}_j)$, $D = D_1 + \cdots + D_n$, and we assume without loss of generality that $\mathfrak{m}_j$ is monic. Then $\mathbf{p}$ is a solution of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ if and only if $\mathbf{p}$ is a relation of $\mathrm{Syz}_{\mathbf{C}}(\mathbf{E})$, where $\mathbf{C} \in \mathbb{K}^{D \times D}$ is $\mathrm{diag}(\mathbf{C}_1, \ldots, \mathbf{C}_n)$ with $\mathbf{C}_j$ the companion matrix associated with $\mathfrak{m}_j$, and $\mathbf{E} \in \mathbb{K}^{m \times D}$ is $[\mathbf{E}_1 | \cdots | \mathbf{E}_n]$ where the column k of the matrix $\mathbf{E}_j \in \mathbb{K}^{m \times D_j}$ is the column vector in $\mathbb{K}^{m \times 1}$ formed by the coefficients of $\mathbf{F}_{*,j}$ of degree k.

In particular, the multiplication $\mathbf{p} \cdot \mathbf{F}$ defined by $\mathbf{C}$ as in Definition 2.4 corresponds to the modular product $\mathbf{p} \cdot \mathbf{F} = \mathbf{p}\mathbf{F} \bmod \mathfrak{M}$. In our algorithms, this will allow us to compute residuals by simply computing $\mathbf{p}\mathbf{F}$ and reducing its entries modulo $\mathfrak{M}$; this is efficient thanks to the assumption $n \in \mathcal{O}(m)$.

2.5.2 Computing shifted Popov solution bases

An overview of known fast algorithms for Problem 9 is given in Table 2.1. Concerning interpolant bases and approximant bases where the multiplication matrix is in Jordan form, we have seen above in Theorems 2.17 and 2.20 that there are deterministic algorithms to compute the $\mathbf{s}$-Popov basis using $\tilde{\mathcal{O}}(m^{\omega-1}D)$ operations, with $D =$

Table 2.1: Fast algorithms for systems of linear modular equations over $\mathbb{K}[X]$ (Problems 9 and 10). In this table, $n \in \mathcal{O}(m)$; *partial* $\mathbf{s}$-*min.* = returns small degree rows of an $\mathbf{s}$-minimal solution basis; *split* = product of known linear factors; $\mathcal{H}_{\mathbf{s},1} : |\mathbf{s} - \min(\mathbf{s})| \leqslant D$; $\mathcal{H}_{\mathbf{s},2} : |\max(\mathbf{s}) - \mathbf{s}| \leqslant D$.

Reference	Cost bound	Moduli	Output	Assumptions
Beckermann [Bec92]	$\mathcal{O}(m^2D^2)$	split	$\mathbf{s}$-minimal	$n = 1$
Van Barel-Bultheel [VBB92]	$\mathcal{O}(m^2D^2)$	split	$\mathbf{s}$-minimal	
Beckermann-Labahn [BL94]	$\mathcal{O}(m^2D^2)$	$X^{D/n}$	$\mathbf{s}$-minimal	
Beckermann-Labahn [BL94]	$\tilde{\mathcal{O}}(m^\omega D)$	$X^{D/n}$	$\mathbf{s}$-minimal	
Beckermann-Labahn [BL00]	$\mathcal{O}(mD^2)$	split	$\mathbf{s}$-Popov	
Giorgi et al. [GJV03]	$\tilde{\mathcal{O}}(m^\omega D/n)$	$X^{D/n}$	$\mathbf{s}$-minimal	
Storjohann [Sto06]	$\tilde{\mathcal{O}}(m^\omega \lceil D/m \rceil)$	$X^{D/n}$	partial $\mathbf{s}$-min.	$\mathcal{H}_{\mathbf{s},1}$
Zhou-Labahn [ZL12]	$\tilde{\mathcal{O}}(m^\omega \lceil D/m \rceil)$	$X^{D/n}$	$\mathbf{s}$-minimal	$\mathcal{H}_{\mathbf{s},1}$ or $\mathcal{H}_{\mathbf{s},2}$
Theorem 2.25 and [CJN+15]	$\tilde{\mathcal{O}}(m^{\omega-1}D)$, probabilistic	any	small solution vector	
Theorem 2.19 and [JNSV17]	$\tilde{\mathcal{O}}(m^{\omega-1}D)$	split	$\mathbf{s}$-minimal	$\mathcal{H}_{\mathbf{s},1}$
Theorem 2.12 and [JNSV17]	$\tilde{\mathcal{O}}(mD^{\omega-1})$	any	$\mathbf{s}$-Popov	$D \in \mathcal{O}(m)$
Theorem 2.20 and [JNSV16]	$\tilde{\mathcal{O}}(m^{\omega-1}D)$	split	$\mathbf{s}$-Popov	
Theorem 2.22 and [Nei16]	$\tilde{\mathcal{O}}(m^{\omega-1}D)$	any	$\mathbf{s}$-Popov	

$\deg(\mathfrak{m}_1) + \cdots + \deg(\mathfrak{m}_n)$. Here, we extend this result to arbitrary moduli, up to the assumption that the number of equations is not much larger than the number of unknowns. The next result follows from Proposition 8.17 and the corresponding Algorithm 18.

Theorem 2.22. *Assuming $n \in \mathcal{O}(m)$, there is a deterministic algorithm which solves Problem 9 using $\tilde{\mathcal{O}}(m^{\omega-1}D)$ operations in $\mathbb{K}$ and returns the $\mathbf{s}$-Popov solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$, where $D = \deg(\mathfrak{m}_1) + \cdots + \deg(\mathfrak{m}_n)$.*

In the particular cases of approximant and interpolant bases, we managed to avoid such an assumption on n and m; more insight into this difference will be given below in the overview of our algorithm. Although this assumption will always be satisfied in this thesis, where we will rely on solution bases to compute shifted normal forms of polynomial matrices and in the Coppersmith technique over $\mathbb{K}[X]$ (see Sections 3.1.5 and 3.2.2), it might be a hindrance in some cases. We remark for example that in Section 3.1.3, we will see some multivariate interpolation problems which reduce to interpolant bases with possibly significantly more equations than unknowns.

In terms of the computation of relation bases as in Problem 4, this directly translates as the following result.

Corollary 2.23. *Let $\mathbf{C} \in \mathbb{K}^{D\times D}$ be a companion-block diagonal matrix with n blocks as in Eq. (2.4). Assuming $n \in \mathcal{O}(m)$, there is a deterministic algorithm which solves Problem 4 with $\mathbf{M} = \mathbf{C}$ using $\tilde{\mathcal{O}}(m^{\omega-1}D)$ operations in $\mathbb{K}$, and returns the basis in $\mathbf{s}$-Popov form.*

We note that this follows directly from Theorem 2.12 when $D \in \mathcal{O}(m)$. If some of the moduli have small degree, or in other words some of the diagonal blocks have small dimension, we use this result for base cases of our recursive algorithm.

As we have discussed in Section 2.1.3, compared to interpolant bases, where the multiplication matrix is in Jordan form, the situation here is quite different in terms of computations since the multiplication matrix $\mathbf{C}$ is not in upper triangular form. We recall that having an arbitrary upper triangular matrix implies that one can then rely on recurrence equations to solve the problem iteratively, using [BL00, Algorithm FFFG] that we recall with our notation in Algorithm 8. This recurrence is used for example in the iterative algorithms of [VBB91, Bec92, VBB92, BL94] for approximant or interpolant bases. Beyond the techniques used to achieve efficiency, the fast algorithms in [BL94, GJV03, ZL12], as well as ours presented in Sections 2.3 and 2.4, are essentially divide-and-conquer versions of this iterative solution and are thus based on the same recurrence equations.

However, for a multiplication matrix that has companion blocks and thus is not upper triangular, there is no such recurrence in general. To give a simple illustration of this difficulty, let us consider the case of one equation modulo $X^D - 1$, that is, the case of the companion multiplication matrix

$$\mathbf{C} = \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & 0 & 1 \\ 1 & 0 & \cdots & 0 \end{bmatrix}.$$

It is not clear how one could split this matrix into submatrices that would correspond to well-defined subproblems that ask to compute relation bases for a multiplication matrix in companion form. In terms of polynomial equations, if our input is $\mathbf{F} \in \mathbb{K}[X]^{m\times 1}$ of degree less than D, it is not clear either what subproblem could be considered whose solutions would represent some progress towards vanishing modulo $X^D - 1$.

Then, a natural idea is to relate solution bases to kernel bases, which leads us to introduce an additional unknown: the matrix of quotients. Indeed, Problem 9 asks to find $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ for which there is some quotient matrix $\mathbf{Q} \in \mathbb{K}[X]^{m\times n}$ such that

$$\begin{bmatrix}\mathbf{P} & \mathbf{Q}\end{bmatrix} \mathbf{V} = 0, \quad \text{where } \mathbf{V} = \begin{bmatrix} \mathbf{F} \\ -\mathrm{diag}(\mathfrak{M}) \end{bmatrix} \in \mathbb{K}[X]^{(m+n)\times n}.$$

More precisely, we will show in Section 8.1 that $\begin{bmatrix}\mathbf{P} & \mathbf{Q}\end{bmatrix}$ can be obtained as a $\mathbf{u}$-Popov kernel basis for $\mathbf{V}$ for the shift $\mathbf{u} = (\mathbf{s} - \min(\mathbf{s}), \mathbf{0}) \in \mathbb{Z}_{\geqslant 0}^{m+n}$. To the best of our knowledge, it is not known how to compute such a kernel basis efficiently; the algorithms in [ZLS12, ZL13] are only efficient for specific families of shifts satisfying assumptions similar to $\mathcal{H}_{\mathbf{s},1}$ and $\mathcal{H}_{\mathbf{s},2}$, and they do not return the shifted Popov basis. In fact, an immediate difficulty lies in the degrees in $\mathbf{Q}$, which may be so large that the size of $\mathbf{Q}$ exceeds our target cost; we recall that we are only interested in the part $\mathbf{P}$ of the kernel $\begin{bmatrix}\mathbf{P} & \mathbf{Q}\end{bmatrix}$.

Yet, when the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ is known, this kernel basis point of view is fruitful. It helps us to reduce Problem 9 with known $\boldsymbol{\delta}$ to the computation of shifted Popov kernel bases with known information on the output pivot degrees and indices;

in Section 8.1, we use our results on the computation of approximant bases to give a fast algorithm for finding a kernel basis when such information is available. For further efficiency, the fact that we are not interested in $\mathbf{Q}$ allows us to use partial linearization so as to reduce the degrees in the computed kernel basis. We thus obtain the desired cost bound when $\boldsymbol{\delta}$ is known; we note that this is a slight generalization of results already obtained in [GS11, Section 3] in the context of Hermite form computation.

Then, for $n \geqslant 1$, our algorithm in Section 8.3 uses a divide-and-conquer approach on the number n of equations, based on the transitivity result in Theorem 1.28. To summarize, this consists in computing a solution basis $\mathbf{P}^{(1)}$ for the first $n/2$ equations; then computing the last $n/2$ columns of the residual $\mathbf{G} = \mathbf{P}^{(1)}\mathbf{F} \bmod \mathfrak{M}$; and finally computing a solution basis $\mathbf{P}^{(2)}$ for $\mathbf{G}$ and the second half of the moduli. As discussed for approximant and interpolant bases above, if one computes $\mathbf{P}^{(1)}$ and $\mathbf{P}^{(2)}$ recursively in shifted Popov form, their product $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$ is an $\mathbf{s}$-minimal solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ which is usually not in $\mathbf{s}$-Popov form and which may have size beyond our target cost. Thus, we avoid the computation of this product by using $\mathbf{P}^{(2)}$ and $\mathbf{P}^{(1)}$ to deduce the $\mathbf{s}$-minimal degree $\boldsymbol{\delta}$, from which we efficiently find the sought $\mathbf{s}$-Popov solution basis as sketched in the previous paragraph.

The main remaining difficulty is to deal with the base case $n = 1$ of the recursion. If this corresponds to an equation whose modulus has small degree, as explained above we can directly rely on the general relation basis algorithm presented in Section 2.2. However, in the case of a single equation whose modulus has large degree D compared to m, it seems that new ingredients are needed. In Section 8.2, we develop the ingredients presented in the next paragraphs, leading to an $\tilde{\mathcal{O}}(m^{\omega-1}D)$ algorithm to solve this base case and thus completing our fast algorithm for Problem 9.

When $n = 1$, we have one equation $\mathbf{F} \in \mathbb{K}[X]^{m\times 1}$ of degree less than D, and a modulus $\mathfrak{m} \in \mathbb{K}[X]_{\neq 0}$ of degree D. From the discussion above, computing the $\mathbf{s}$-Popov solution basis $\mathbf{P}$ for $\mathrm{Syz}_{\mathfrak{m}}(\mathbf{F})$ amounts to computing the $\mathbf{u}$-Popov kernel basis $\begin{bmatrix}\mathbf{P} & \mathbf{Q}\end{bmatrix}$ of the column $\mathbf{V} = [\mathbf{F}^\mathsf{T} \;\; \mathfrak{m}]^\mathsf{T}$, where $\deg(\mathbf{F}) < \deg(\mathfrak{m})$ and the last entry of $\mathbf{u}$ is $\min(\mathbf{u})$. Our ingredients give a central role to a quantity that we have called the *amplitude* of the shift $\mathbf{u}$ and which is defined as $\mathrm{amp}(\mathbf{u}) = \max(\mathbf{u}) - \min(\mathbf{u})$.

First, when $\mathrm{amp}(\mathbf{u}) \in \mathcal{O}(D)$ we show that the $\mathbf{u}$-Popov kernel basis $\begin{bmatrix}\mathbf{P} & \mathbf{Q}\end{bmatrix}$ can be efficiently obtained as a submatrix of the $\mathbf{u}$-Popov approximant basis for the matrix $\mathbf{V}$ and an order in $\mathcal{O}(D)$. Then, when $\mathrm{amp}(\mathbf{u})$ is large compared to D and assuming that $\mathbf{u}$ is sorted non-decreasingly, $\mathbf{P}$ has a lower block triangular shape. We show how the size of these blocks can be revealed, along with the $\mathbf{s}$-pivot degree of $\mathbf{P}$, using a divide-and-conquer approach which splits $\mathbf{u}$ into two shifts of amplitude about $\max(\mathbf{u})/2$.

Remark 2.24. In our algorithm, the assumption $n \in \mathcal{O}(m)$ is required for the efficiency of two operations.

In the situation where we know the $\mathbf{s}$-minimal degree $\boldsymbol{\delta}$, we rely on the computation of a kernel basis for the $(m+n)\times n$ matrix $\mathbf{V}$, for which our target cost currently seems out of reach when $n \gg m$. To solve this, one may try to exploit the structure of $\mathbf{V}$, whose bottom part is an $n \times n$ diagonal matrix, or to find another way to solve the problem when $\boldsymbol{\delta}$ is known.

Our algorithm to compute the residual $\mathbf{G} = \mathbf{P}^{(1)}\mathbf{F} \bmod \mathfrak{M}$ is not efficient when $\mathfrak{M}$

contains many polynomials of small degree, that is, $n \gg m$. In the case of interpolant bases, we have solved this in Section 14.2 by using the known linear factors of the moduli, which allow us to group together small degree equations into fewer equations of larger degree via Hermite interpolation and evaluation. Generalizing this to arbitrary moduli, using in particular Chinese remainder evaluation and interpolation, might lead to the desired cost bound but has not been studied in this thesis. ☕

2.5.3 Computing a solution via structured linear algebra

In several situations, such as applications to list-decoding algorithms (see Section 3.1.2), one is not interested in computing a *basis* of solutions which is *minimal* for the shifted degree, but rather in finding *one* solution of *sufficiently small* shifted degree.

More precisely, given the moduli $\mathfrak{M}$, the system matrix $\mathbf{F}$, and prescribed degree constraints specified by m positive integers $N_1, \ldots, N_m$, one looks for a solution $\mathbf{p} = [p_j]_j$ for $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ such that $\deg(p_j) < N_j$ for all j. We formalize this question in Problem 10.

Problem 10 – *Small solution vector*

Input:

- polynomials $\mathfrak{M} = (\mathfrak{m}_1, \ldots, \mathfrak{m}_n) \in \mathbb{K}[X]^n_{\neq 0}$,
- a matrix $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ such that $\deg(\mathbf{F}_{*,j}) < \deg(\mathfrak{m}_j)$,
- positive integers $N_1, \ldots, N_m$.

Output:

- a row vector $\mathbf{p} = [p_1, \ldots, p_m] \in \mathbb{K}[X]^{1 \times m}$ such that
  - $\mathbf{p}$ is a solution of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$,
  - $\deg(p_j) < N_j$ for all $1 \leqslant j \leqslant m$.

We recall from Remark 1.9 that degree constraints are directly related to shifted degrees, since $\deg(p_j) < N_j$ for $1 \leqslant j \leqslant m$ is equivalent to $\mathrm{rdeg}_{-\mathbf{N}}(\mathbf{p}) < 0$ with $\mathbf{N}$ the shift $\mathbf{N} = (N_1, \ldots, N_m) \in \mathbb{Z}^m_{>0}$. Therefore, to solve this problem one may directly apply a solution basis algorithm and return one of the rows of the output basis which satisfy $\mathrm{rdeg}_{-\mathbf{N}}(\mathbf{p}) < 0$; if there is no such row, this means that there is no solution to the given instance of Problem 10.

Another approach is proposed in Chapter 9, where we give two algorithms which solve Problem 10 relying on structured $\mathbb{K}$-linear algebra. Using the currently fastest known algorithms for solving structured linear systems, this provides a solution to Problem 10 with fewer logarithmic terms in the cost bound than relying on our solution basis algorithm presented in Section 2.5.2.

Theorem 2.25. *There is a probabilistic algorithm that either computes a solution to Problem 10, or determines that none exists, using*

$$\mathcal{O}((m+n)^{\omega-1}\mathsf{M}(D)\log(D))$$

operations in $\mathbb{K}$, where $D = \deg(\mathfrak{m}_1) + \cdots + \deg(\mathfrak{m}_n)$. Both Algorithms 19 and 20 achieve this result. These algorithms choose $\mathcal{O}(D)$ elements in $\mathbb{K}$; if these elements are chosen

uniformly at random in a subset $\Gamma \subseteq \mathbb{K}$ *of cardinality at least* $6(D+1)^2$*, then the probability of success is at least* $1/2$.

Our algorithms involve two different linearizations of the system of linear modular equations over $\mathbb{K}[X]$ into a homogeneous linear system over $\mathbb{K}$. Defining $N = N_1 + \cdots + N_m$, both systems have D equations in N unknowns. Furthermore, the structure of the latter systems allows us to solve them efficiently using the algorithms in [BJS08, BJMS16][3]. While it is customary to make the assumption $N > D$ to ensure that our problem admits a solution, here we do not need and thus do not make this assumption, since the algorithm will detect whether no solution exists.

Our first algorithm, detailed in Section 9.2, first rewrites the equations with arbitrary moduli into equations with moduli that are powers of X, by means of reversing the order of the coefficients of the involved polynomials. This transforms the problem into an approximant basis problem, with the number of unknowns being increased from m to $m+n$ and the sum of the degrees of the moduli being now $D+n(\max(\mathbf{N})-1)$. According to Section 2.3, computing a minimal approximant basis for such input dimensions can be done in $\tilde{\mathcal{O}}((m+n)^{\omega-1}(D+n\max(\mathbf{N})))$ operations with the fastest known algorithms; for some $\mathbf{N}$ this exceeds the cost in the theorem above. Here, noticing that we are not interested in computing the n unknowns that were added in the transformation, we manage to derive a structured $\mathbb{K}$-linear system which can be solved in $\tilde{\mathcal{O}}((m+n)^{\omega-1}D)$ operations. The matrix of the system is mosaic-Hankel, meaning that it is formed by Hankel blocks of various dimensions.

This transformation to an approximant basis problem was inspired by, and generalizes, the derivation of the *extended key equations* in the context of decoding algorithms in coding theory. These equations extend the key equation designed for the decoding of BCH and Reed-Solomon codes [Ber68, Ch. 7] (see also [Fit95]), to the more general list-decoding of Reed-Solomon codes. They were first introduced by Roth and Ruckenstein in [RR00] in the context of the Sudan algorithm [Sud97], and then in a more general form by Zeh, Gentner, and Augot [ZGA11] for the Guruswami-Sudan algorithm [GS99]. In our first algorithm, the derivation of the key equation follows that in [ZGA11].

In our second algorithm, detailed in Section 9.3, the homogeneous linear system over $\mathbb{K}$ is directly obtained from the polynomial system matrix $\mathbf{F}$ and the degree contraints $\mathbf{N}$. This linearization is a slight variation of the striped Krylov linearization in [BL00] and Section 4.1 in the case of a companion-block diagonal multiplication matrix. In this case, the structure of the system is less eye-catching than in the case of mosaic-Hankel matrices in the sense that one cannot straightforwardly observe it by just reading the coefficients. Yet, we prove that, thanks to the fact that the multiplication matrix is companion-block diagonal, the system has a *Toeplitz-like* structure with *displacement rank* $m+n$ (these notions are recalled in Section 9.1). Furthermore, we can efficiently compute generators that give us a compact representation of the system. This again allows us to find a solution in $\tilde{\mathcal{O}}((m+n)^{\omega-1}D)$ operations, by resorting to the algorithm of [BJS08, BJMS16].

In short, both points of view lead to the same complexity result, stated in Theorem 2.25 above. In this result, the probability analysis is a standard consequence of the Zippel-

[3] At the time of writing this document, [BJMS16] has not yet been published. In short, for the uses we make of it, the main difference with [BJS08] is that it removes a logarithm factor in the cost bound.

Schwartz lemma; as usual, the probability of success can be made arbitrarily close to one by increasing the size of the subset Γ. If the field $\mathbb{K}$ has fewer than $6(D+1)^2$ elements, then a probability of success at least $1/2$ can still be achieved by using a field extension $\mathbb{L}$ of degree $d \in \mathcal{O}(\log_{\mathrm{Card}(\mathbb{K})}(D))$, up to a cost increase by a factor in $\mathcal{O}(\mathsf{M}(d)\log(d))$.

Specifically, one can proceed in three steps. First, we take $\mathbb{L} = \mathbb{K}[X]/\langle f \rangle$ with $f \in \mathbb{K}[X]$ irreducible of degree d; such an f can be set up using an expected number of $\mathcal{O}\tilde{}(d^2) \subseteq \mathcal{O}(D)$ operations in $\mathbb{K}$ [GG13, Section 14.9]. Then we solve Problem 10 over $\mathbb{L}$ by means of the algorithm of Theorem 2.25, thus using $\mathcal{O}(\rho^{\omega-1}\mathsf{M}(D)\log(D) \cdot \mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$. Finally, from this solution over $\mathbb{L}$ one can deduce a solution over $\mathbb{K}$ using $\mathcal{O}(Dd)$ operations in $\mathbb{K}$. This last point comes from the fact that, as said above, Problem 10 amounts to finding a nonzero vector $\mathbf{u}$ over $\mathbb{K}$ such that $\mathbf{A}\mathbf{u} = 0$ for some matrix $\mathbf{A} \in \mathbb{K}^{D \times (D+1)}$: once we have obtained a solution $\mathbf{v}$ over $\mathbb{L}$, it thus suffices to rewrite it as $\mathbf{v} = \sum_{0 \leqslant i < d} \mathbf{v}_i x^i \neq 0$ (where x is the image of X in $\mathbb{L}$) and, noting that $\mathbf{A}\mathbf{v}_i = 0$ for all i, to find a nonzero $\mathbf{v}_i$ in $\mathcal{O}(Dd)$ comparisons with zero and return it as a solution over $\mathbb{K}$.

3

Impact on related problems

3.1 Multivariate interpolation and list-decoding algorithms

In this section, we apply our algorithms for solution bases and interpolant bases (Sections 2.4 and 2.5) to efficiently solve problems of constrained multivariate interpolation with multiplicities. These problems arise in particular in coding theory, and more specifically in list-decoding algorithms. Our contributions lead to the best known cost bound we are aware of for bivariate interpolation problems, such as those appearing in the list-decoding [Sud97, GS98] and the soft-decoding [KV03a, Sec. III] of Reed-Solomon codes; and for multivariate interpolation problems, such as those appearing in the list-decoding of folded Reed-Solomon codes [GR08] and in robust Private Information Retrieval [DGH12].

In some of these contexts, the input points have a property that allows us to reduce the interpolation problem to a system of linear modular equations which involves less equations than unknowns. Such a system can be solved efficiently with our algorithm for finding solutions with degrees constraints (Section 2.5.3). This interpolation problem and the corresponding list-decoding contexts are presented in Sections 3.1.1 and 3.1.2.

In other situations, this property may not be satisfied. After reduction, the system may have significantly more equations than unknowns, which makes the algorithm of Section 2.5.3 inefficient. To solve this issue, we rather rely on our algorithms for interpolant bases (Section 2.4): while slightly slower, they make no requirement on the input points. In Section 3.1.3, we present this more general multivariate interpolation and the reduction to interpolant bases; consequences for decoding algorithms are given in Section 3.1.4.

Finally, in Section 3.1.5 we focus on the Coppersmith technique over $\mathbb{K}[X]$, which is a generalization of the list-decoding problem for Reed-Solomon codes. Extending our work on the Guruswami-Sudan algorithm, we show that the interpolation step of this technique can be efficiently reduced to a system of linear modular equations.

3.1.1 Multivariate interpolant with degree constraints

Here, we consider a multivariate interpolation problem with multiplicities and degree constraints (Problem 11 below). As mentioned above, one motivation for studying this

problem lies in its applications to list-decoding algorithms. In this context, the parameters r, λ, ν, b of the problem are respectively known as the number of variables, list size, code length or number of interpolation points, and as an agreement parameter. Furthermore, the integers $\mu_1, \ldots, \mu_\nu$ are known as multiplicities associated with each of the ν points, and the r variables are associated with some weights $w_1, \ldots, w_r$.

For example, in the application to the list-decoding of Reed-Solomon codes detailed in Section 3.1.2, we have $r = 1$, all the multiplicities are equal to a same value μ, $\nu - b/\mu$ is an upper bound on the number of errors allowed on a received word, and the weight $w := w_1$ is such that $w + 1$ is the dimension of the code.

We stress that here we do not address the issue of choosing the parameters r, λ, $\mu_1, \ldots, \mu_\nu$ with respect to ν, b, $w_1, \ldots, w_r$, as is often done: in our context, these are all input parameters. Similarly, although we will mention them, we do not make some usual assumptions on these parameters; in particular, we do not make any assumption that ensures that our problem admits a solution: the algorithm will detect whether no solution exists.

In what follows, we have r variables $Y_1, \ldots, Y_r$ and our problem asks to find a polynomial in $\mathbb{K}[X, Y_1, \ldots, Y_r]$. We write $\deg_{Y_1,\ldots,Y_r}$ for the total degree with respect to these variables $Y_1, \ldots, Y_r$. Furthermore, $\mathrm{wdeg}_{w_1,\ldots,w_r}$ stands for the *weighted degree* with respect to weights $w_1, \ldots, w_r \in \mathbb{Z}$ on variables $Y_1, \ldots, Y_r$, respectively; that is, for a polynomial $Q = \sum_{j_1,\ldots,j_r} Q_{j_1,\ldots,j_r}(X) Y_1^{j_1} \cdots Y_r^{j_r}$,

$$\mathrm{wdeg}_{w_1,\ldots,w_r}(Q) \;=\; \max_{j_1,\ldots,j_r} \big(\deg(Q_{j_1,\ldots,j_r}) \;+\; j_1 w_1 + \cdots + j_r w_r\big).$$

Problem 11 – *Constrained multivariate interpolation*

Input:

- points $\{(x_k, y_{k,1}, \ldots, y_{k,r}) \in \mathbb{K}^{r+1}, 1 \leqslant k \leqslant \nu\}$ with pairwise distinct x_k,
- multiplicities $\mu_1, \ldots, \mu_\nu \in \mathbb{Z}_{>0}$,
- degree constraints $\lambda \in \mathbb{Z}_{>0}$ and $b \in \mathbb{Z}$,
- weights $w_1, \ldots, w_r \in \mathbb{Z}$.

Output:

- a nonzero polynomial $Q \in \mathbb{K}[X, Y_1, \ldots, Y_r]$ such that
  - *(i)* $\deg_{Y_1,\ldots,Y_r}(Q) \leqslant \lambda$,
  - *(ii)* $\mathrm{wdeg}_{w_1,\ldots,w_r}(Q) < b$,
  - *(iii)* $Q(x_i, y_{i,1}, \ldots, y_{i,r}) = 0$ with multiplicity at least μ_i for $1 \leqslant i \leqslant \nu$.

We call conditions *(i)*, *(ii)*, and *(iii)* the *list-size* condition, the *weighted degree* condition, and the *vanishing* condition. Hereafter, a point $(x, y_1, \ldots, y_r)$ is said to be a zero of Q *with multiplicity at least* μ if the shifted polynomial $Q(X + x, Y_1 + y_1, \ldots, Y_r + y_r)$ has no monomial of total degree less than μ; in characteristic zero or larger than μ, this is equivalent to requiring that all the derivatives of Q of order up to $\mu - 1$ vanish at $(x, y_1, \ldots, y_r)$.

By linearizing condition *(iii)* under the assumption that conditions *(i)* and *(ii)* are satisfied, it is easily seen that solving Problem 11 amounts to computing a nonzero solution to a $D \times N$ homogeneous linear system over $\mathbb{K}$. Here, the number D of equations derives from condition *(iii)* and thus depends on r, ν, $\mu_1, \ldots, \mu_\nu$, while the number N of unknowns derives from conditions *(i)* and *(ii)* and thus depends on r, λ, b, $w_1, \ldots, w_r$. It is customary to assume $D < N$ in order to guarantee the existence of a nonzero solution; however, as said above, we do not make this assumption, since our algorithms do not require it.

Problem 11 is a generalization of the interpolation step of the Guruswami-Sudan algorithm [Sud97, GS99] to r variables $Y_1, \ldots, Y_r$, distinct multiplicities, and distinct weights. The multivariate case $r > 1$ occurs for instance in the list-decodings of Parvaresh-Vardy codes [PV05] and folded Reed-Solomon codes [GR08]. We will discuss these specific contexts in Section 3.1.2. The case of distinct multiplicities occur in the interpolation step in soft-decoding of Reed-Solomon codes [KV03a]; however, this context differs from Problem 11 in that the x_i are not necessarily pairwise distinct, a situation that we will thus study separately in Sections 3.1.3 and 11.2.4.

Our solution to Problem 11 relies on a reduction to the problem of solving a system of linear modular equations over $\mathbb{K}[X]$ (Problem 10). This is detailed in Section 11.1 and is essentially based on a rewriting of the vanishing condition as a set of divisibility properties of the successive *Hasse derivatives* of the sought multivariate polynomial, extending to the multivariate case $r > 1$ results that can be found in [ZGA11] and [Zeh13, Chapters 4 and 5]. In this reduction, the assumption that $x_1, \ldots, x_\nu$ are distinct is a key to obtain an instance of Problem 10 that we know how to solve efficiently (Theorem 2.25). Then, we immediately deduce the following cost bound for Problem 11; in this result, m corresponds to the number of polynomial unknowns of the obtained instance of Problem 10, while n corresponds to the number of equations.

Corollary 3.1. *Let*

$$\mathcal{S} = \Big\{ (j_1, \ldots, j_r) \in \mathbb{Z}_{\geqslant 0}^r \;\Big|\; j_1 + \cdots + j_r \leqslant \lambda \;\; \textit{and} \;\; j_1 w_1 + \cdots + j_r w_r < b \Big\},$$

and let $\mu = \max_{1 \leqslant i \leqslant \nu} \mu_i$, $m = \mathrm{Card}(\mathcal{S})$, $n = \binom{r+\mu-1}{r}$, *and* $D = \sum_{1 \leqslant i \leqslant \nu} \binom{r+\mu_i}{r+1}$. *There exists a probabilistic algorithm that either computes a solution to Problem 11, or determines that none exists, using*

$$\mathcal{O}((m+n)^{\omega-1} \mathsf{M}(D) \log(D)) \subseteq \mathcal{O}\tilde{}((m+n)^{\omega-1} D)$$

operations in $\mathbb{K}$. *This can be achieved using Algorithm 23 followed by Algorithm 19 or 20. These algorithms choose* $\mathcal{O}(D)$ *elements in* $\mathbb{K}$*; if these elements are chosen uniformly at random in a subset of* $\mathbb{K}$ *of cardinality at least* $6(D+1)^2$*, then the probability of success is at least* $1/2$.

If $\mathbb{K}$ has fewer than $6(D+1)^2$ elements, the remarks made after Theorem 2.25 about working in a field extension of $\mathbb{K}$ still apply here. Furthermore, since in Problem 11 the points x_i are assumed to be distinct, we have already $\mathrm{Card}(\mathbb{K}) \geqslant \nu$ and then we can take an extension of degree $d = \mathcal{O}(\log_\nu(D))$. In all the applications to error-correcting codes

we will consider (see Section 3.1.2), D is polynomial in ν so that we can take $d = \mathcal{O}(1)$, and therefore in this context the cost bound in Corollary 3.1 holds for any field $\mathbb{K}$.

If one is concerned about the probabilistic aspect of this result, we note that another algorithm is proposed in Corollary 3.2; while slower by some logarithmic factors, it is deterministic and returns a whole shifted reduced basis of interpolants in the slightly more general context of Problem 12.

For a comparison with previous work, since Problem 11 directly reduces to Problems 8 to 10, we refer the reader to the discussions in the corresponding Sections 2.4 and 2.5. In the next section, we give more details about previous work on the specific case of the interpolation steps in list-decoding algorithms for Reed-Solomon and folded Reed-Solomon codes.

We will often refer to the two following assumptions on the input parameters:

$\mathcal{H}_{\text{int},1}$: $\mu \leqslant \lambda$,

$\mathcal{H}_{\text{int},2}$: $b > 0$ and $b > \lambda \cdot \max_{1 \leqslant j \leqslant r} w_j$.

Regarding $\mathcal{H}_{\text{int},1}$, we prove in Section 11.4 that the case $\mu > \lambda$ can be reduced to the case $\mu = \lambda$, so that this assumption can be made without loss of generality. Besides, it is easily verified that $\mathcal{H}_{\text{int},2}$ is equivalent to having $\mathcal{S} = \{(j_1, \ldots, j_r) \in \mathbb{Z}_{\geqslant 0}^r \mid j_1 + \cdots + j_r \leqslant \lambda\}$; when $w_j > 0$ for some j, $\mathcal{H}_{\text{int},2}$ means that we do not take λ uselessly large. In particular, assuming $\mathcal{H}_{\text{int},1}$ and $\mathcal{H}_{\text{int},2}$ implies that $m = \text{Card}(\mathcal{S}) = \binom{r+\lambda}{r} \geqslant n$.

3.1.2 List-decoding of (folded) Reed-Solomon codes

We now focus on the specific context of the interpolation steps of the Guruswami-Sudan algorithm [Sud97, GS99] for the list-decoding of Reed-Solomon codes, and the Guruswami-Rudra algorithm [GR08] for the list-decoding of folded Reed-Solomon codes. We compare our results with algorithms designed for these situations; for a broader overview of previous work, we refer the reader to the introductive section of [BB10] and to [Nie13, Section 3.6].

In the case of the Guruswami-Sudan interpolation step, we have $r = 1$ and the assumptions $\mathcal{H}_{\text{int},1}$ and $\mathcal{H}_{\text{int},2}$ are satisfied as well as

$\mathcal{H}_{\text{int},3}$: $0 \leqslant w < \nu$ where $w := w_1$,

$\mathcal{H}_{\text{int},4}$: $\mu_1 = \cdots = \mu_\nu = \mu$.

The assumption $\mathcal{H}_{\text{int},3}$ corresponds to the coding theory context, where $w + 1$ is the dimension of the code; then $w + 1$ must be positive and at most ν, which is the length of the received word. To support this assumption independently from any application context, we show in Section 11.5 that if $w \geqslant \nu$, then Problem 11 has either a trivial solution or no solution at all.

Under the assumptions $\mathcal{H}_{\text{int},1}$, $\mathcal{H}_{\text{int},2}$, $\mathcal{H}_{\text{int},3}$, and $\mathcal{H}_{\text{int},4}$, the quantities defined in Corollary 3.1 are $m = \text{Card}(\mathcal{S}) = \lambda + 1$, $n = \mu$, and $D = \frac{\mu(\mu+1)}{2}\nu$.

Our contributions to this decoding problem are detailed in Section 11.2; in short, we will show the following consequences of Corollary 3.1:

- the interpolation step of the Guruswami-Sudan algorithm [GS99] can be performed in $\tilde{\mathcal{O}}(\lambda^{\omega-1}\mu_{\mathrm{GS}}^2\nu)$ operations, where μ_{GS} is the multiplicity parameter used in this algorithm;

- the interpolation step of the Wu algorithm [Wu08] can be performed in $\tilde{\mathcal{O}}(\lambda^{\omega-1}\mu_{\mathrm{Wu}}^2\nu)$ operations, where μ_{Wu} is the multiplicity parameter used in this algorithm;

- the re-encoding technique [WB86, KV03b] can be used in conjunction with our algorithm in order to reduce the cost of the interpolation step of the Guruswami-Sudan algorithm to $\tilde{\mathcal{O}}(\lambda^{\omega-1}\mu_{\mathrm{GS}}^2(\nu-w))$ operations.

To the best of our knowledge, these are the best known cost bounds for these tasks. If the probabilistic aspect of the algorithm is an issue, the same cost bounds can be obtained, up to an additional logarithmic factor, by the deterministic algorithm presented in Section 3.1.3 (see Corollary 3.2).

Furthermore, our result can also be adapted to the context of the interpolation step of the Kötter-Vardy algorithm for the soft-decoding of Reed-Solomon codes [KV03a]. Still, in this context the field elements $x_1, \ldots, x_\nu$ are generally not pairwise distinct, implying that after our reduction we obtain an instance of Problem 10 which we may not know how to solve efficiently; the reason is that the number n of polynomial equations may be too large compared to the number m of unknowns. For this particular situation as well, we refer the reader Section 3.1.3 for a fast algorithm.

Previous results focus mostly on the Guruswami-Sudan case ($r = 1, \mu \geqslant 1$) and some of them more specifically on the Sudan case ($r = \mu = 1$); we summarize these results in Table 3.1. We also include in this table the cost bounds of the algorithms in [VBB92, Bec92] which solve Problem 8, since they can be used via the aforementioned reduction. In some cases, such as in [Rei03, Ale05, Ber11, CH15], the cost bounds were not stated quite exactly in our terms but the translation is straightforward.

In the second column of that table, we give the cost with respect to the interpolation parameters λ, μ, ν, assuming further $\mu = \nu^{\mathcal{O}(1)}$ and $\lambda = \nu^{\mathcal{O}(1)}$ to simplify the logarithmic factors. The most significant part of the running time is its dependency with respect to ν, with results being either cubic, quadratic, or quasi-linear. Then, under the assumption $\mathcal{H}_{\mathrm{int},1}$, the second most important parameter is λ, followed by μ. In particular, our result compares favorably to the cost $\tilde{\mathcal{O}}(\lambda^{\omega}\mu\nu)$ obtained by Cohn and Heninger [CH15] which was, to our knowledge, the best previous bound for this problem.

In the third column, we give the cost with respect to the Reed-Solomon code parameters ν and w, using worst-case parameter choices that are made to ensure the existence of a solution: $\mu = \mathcal{O}(\nu w)$ and $\lambda = \mathcal{O}(\nu^{3/2}w^{1/2})$ in the Guruswami-Sudan case [GS99], and $\lambda = \mathcal{O}(\nu^{1/2}w^{-1/2})$ in the Sudan case [Sud97]. With these parameter choices, our algorithms present a speedup $(\nu/w)^{1/2}$ over the algorithm in [CH15].

Most previous algorithms rely on linear algebra, either over $\mathbb{K}$ or over $\mathbb{K}[X]$. When working over $\mathbb{K}$, a natural idea is to rely on cubic-time general linear system solvers, as in Sudan's and Guruswami-Sudan's original papers.

Another line of work uses faster linear system solvers which are specialized for systems that have specific structures. The algorithms in [RR00, ZGA11] rely on Feng and Tzeng's

Table 3.1: Fast algorithms for the interpolation step of Guruswami-Sudan list-decoding (Problem 11 with $r = 1$ and under the assumptions $\mathcal{H}_{\text{int},1}$, $\mathcal{H}_{\text{int},2}$, $\mathcal{H}_{\text{int},3}$, $\mathcal{H}_{\text{int},4}$). The symbol $\star$ indicates a probabilistic algorithm.

Sudan case ($\mu = 1$)			
Beckermann [Bec92]	$\mathcal{O}(\lambda\nu^2)$	$\mathcal{O}(\nu^{\frac{5}{2}} w^{\frac{-1}{2}})$	
Sudan [Sud97]	$\mathcal{O}(\nu^3)$	$\mathcal{O}(\nu^3)$	
Olshevsky-Shokrollahi [OS99]	$\mathcal{O}(\lambda\nu^2)$	$\mathcal{O}(\nu^{\frac{5}{2}} w^{\frac{-1}{2}})$	
Roth-Ruckenstein [RR00]	$\mathcal{O}(\lambda\nu^2)$	$\mathcal{O}(\nu^{\frac{5}{2}} w^{\frac{-1}{2}})$	
Cohn-Heninger [CH11, CH15]	$\mathcal{O}(\lambda^\omega \mathsf{M}(\nu)\log(\nu))$	$\mathcal{O}\tilde{}(\nu^{\frac{\omega+3}{2}} w^{\frac{-\omega}{2}})$	$\star$
Chapter 11 and Corollary 3.1	$\mathcal{O}(\lambda^{\omega-1} \mathsf{M}(\nu)\log(\nu))$	$\mathcal{O}\tilde{}(\nu^{\frac{\omega+1}{2}} w^{\frac{1-\omega}{2}})$	$\star$
Chapter 13 and Corollary 3.2	$\mathcal{O}(\lambda^{\omega-1} \mathsf{M}(\nu)\log(\nu)^2)$	$\mathcal{O}\tilde{}(\nu^{\frac{\omega+1}{2}} w^{\frac{1-\omega}{2}})$	
Guruswami-Sudan case ($\mu \geqslant 1$)			
Van Barel-Bultheel [VBB92]	$\mathcal{O}(\lambda\mu^4\nu^2)$	$\mathcal{O}(\nu^{\frac{15}{2}} w^{\frac{9}{2}})$	
Guruswami-Sudan [GS99]	$\mathcal{O}(\mu^6\nu^3)$	$\mathcal{O}(\nu^9 w^6)$	
Olshevsky-Shokrollahi [OS99]	$\mathcal{O}(\lambda\mu^4\nu^2)$	$\mathcal{O}(\nu^{\frac{15}{2}} w^{\frac{9}{2}})$	
Zeh-Gentner-Augot [ZGA11]	$\mathcal{O}(\lambda\mu^4\nu^2)$	$\mathcal{O}(\nu^{\frac{15}{2}} w^{\frac{9}{2}})$	
Kötter, Nielsen-Høholdt, McEliece [Köt96, NH00, McE03]	$\mathcal{O}(\lambda\mu^4\nu^2)$	$\mathcal{O}(\nu^{\frac{15}{2}} w^{\frac{9}{2}})$	
Reinhard [Rei03]	$\mathcal{O}(\lambda^3\mu^2\nu^2)$	$\mathcal{O}(\nu^{\frac{17}{2}} w^{\frac{7}{2}})$	
Lee-O'Sullivan [LO08]	$\mathcal{O}(\lambda^4\mu\nu^2)$	$\mathcal{O}(\nu^9 w^3)$	
Trifonov [Tri10] (heuristic)	$\mathcal{O}(\mu^3\nu^2)$	$\mathcal{O}(\nu^5 w^3)$	
Alekhnovich [Ale05]	$\mathcal{O}(\lambda^4\mu^4 \mathsf{M}(\nu)\log(\nu))$	$\mathcal{O}\tilde{}(\nu^{11} w^6)$	
Beelen-Brander [BB10]	$\mathcal{O}(\lambda^3 \mathsf{M}(\lambda\mu\nu)\log(\nu))$	$\mathcal{O}\tilde{}(\nu^8 w^3)$	
Bernstein [Ber11]	$\mathcal{O}(\lambda^\omega \mathsf{M}(\lambda\nu)\log(\nu))$	$\mathcal{O}\tilde{}(\nu^{\frac{3\omega+5}{2}} w^{\frac{\omega+1}{2}})$	$\star$
Cohn-Heninger [CH11, CH15]	$\mathcal{O}(\lambda^\omega \mathsf{M}(\mu\nu)\log(\nu))$	$\mathcal{O}\tilde{}(\nu^{\frac{3\omega+4}{2}} w^{\frac{\omega+2}{2}})$	$\star$
Chapter 11 and Corollary 3.1	$\mathcal{O}(\lambda^{\omega-1} \mathsf{M}(\mu^2\nu)\log(\nu))$	$\mathcal{O}\tilde{}(\nu^{\frac{3\omega+3}{2}} w^{\frac{\omega+3}{2}})$	$\star$
Chapter 13 and Corollary 3.2	$\mathcal{O}(\lambda^{\omega-1} \mathsf{M}(\mu^2\nu)\log(\nu)^2)$	$\mathcal{O}\tilde{}(\nu^{\frac{3\omega+3}{2}} w^{\frac{\omega+3}{2}})$	

linear system solver [FT91], combined with a reformulation in terms of syndromes and key equations. Here, we use (and generalize to the case $r > 1$) some of these results in Section 9.2, and we rely on the more efficient structured linear system solver of [BJS08, BJMS16][4]. Prior to our work, Olshevsky and Shokrollahi also used structured linear algebra techniques [OS99], but it is unclear to us whether their encoding of the problem could lead to similar results as ours.

As said above, another approach rephrases the problem of computing Q in terms of polynomial matrix computations. This is based on the fact that the set of polynomials which satisfy the conditions *(i)* and *(iii)* of Problem 11 form a free $\mathbb{K}[X]$-module of finite rank; precisely, this rank is the quantity $m = \mathrm{Card}(\mathcal{S})$ defined in Corollary 3.1. In the context here we have $m = \lambda + 1$, and there are essentially two approaches.

First, the incremental algorithms of [Köt96, NH00] and [McE03, Section 7], which are often referred to as *Kötter's algorithm* in coding theory, compute a shifted reduced basis of the module by an iteration over the points and over the multiplicity constraints. In fact, these algorithms can be interpreted as particular instances of iterative algorithms for M-Padé approximation, by Beckermann [Bec92] when $\mu = 1$ and by Van Barel and Bultheel when $\mu \geqslant 1$ [VBB92]. More insight into this remark will be given Sections 3.1.3 and 3.1.4, where we also show that in this decoding context the input shift is sufficiently balanced so that this type of algorithms uses $\mathcal{O}(\lambda\mu^4\nu^2)$ operations. (According to Section 2.4.1, for worst-case shifts this approach could use $\mathcal{O}(\lambda^2\mu^4\nu^2)$ operations, unless one normalizes the bases at each iteration as in [BL00, Algorithm FFFG]; see also Section 6.4.)

A second approach consists in starting from known generators of the module and then computing a shifted reduced basis of it; this is the analogue of the Coppersmith technique over $\mathbb{K}[X]$ [Rei03, CH15] (see also Section 3.1.5). The algorithms in [Ale02, Rei03, LO08, Bus08, BB10, Bra10, Ber11, CH11] follow this scheme, which yields a vector which satisfies the degree constraints of condition *(ii)* of Problem 11, unless there is no solution. Here, the entries of the shift are small enough not to impact the cost bound of the fastest known row reduction algorithms, which are designed for the uniform shift. To achieve quasi-linear time in ν, the algorithms in [BB10, Bra10] use a reduction subroutine due to Alekhnovich [Ale05], while those in [Ber11, CH11, CH15] rely on the faster, randomized algorithm in [GJV03]. Up to additional logarithmic factors, or the assumption that $\mathsf{M}(d) \in \mathcal{O}(d^{\omega-1})$, a similarly fast and deterministic algorithm was given in [GSSV12].

The case $r > 1$ is used for example in the interpolation steps of the list-decoding of Parvaresh-Vardy codes [PV05] and folded Reed-Solomon codes [GR08], as well as for robustness issues in Private Information Retrieval [DGH12]. In these contexts, one deals with Problem 11 for some $r > 1$, identical positive weights $w_1 = \cdots = w_r$, and with list-size and multiplicity parameters λ and $\mu = \mu_1 = \cdots = \mu_\nu$ such that the main quantities in the cost bound of Corollary 3.1 are

$$\mathcal{S} = \{j \in \mathbb{Z}^r_{\geqslant 0} \mid |j| < \lambda\},\ m = \binom{r+\lambda}{r},\ n \leqslant m,\ \text{and}\ D = \binom{r+\mu}{r+1}\nu.$$

Two algorithms were first given for this case, both having cost bound $\mathcal{O}(rD^3)$: the first

[4] At the time of writing this document, [BJMS16] has not yet been published. In short, for the uses we make of it, the main difference with [BJS08] is that it removes a logarithm factor in the cost bound.

one generalizes the incremental solution of [Bec92, VBB92] to this multivariate context (see [Fit97, Section 4], [OF02]), while the other one relies on an extension of the FGLM algorithm [FGLM93] to the case of modules of finite codimension (see [Fit97, Sections 2 and 3]).

Then, the approach based on the computation of a reduced basis of a $\mathbb{K}[X]$-module was extended to the multivariate case $r > 1$ in [Bus08, Bra10, CH12]. We give more details about this in Section 11.3; in short, by relying on fast row reduction [GSSV12], this gives a deterministic algorithm which uses $\tilde{\mathcal{O}}(m^\omega \mu\nu)$ operations. To the best of our knowledge, this is the previously best known cost bound for this multivariate context. Our cost bound $\tilde{\mathcal{O}}(m^{\omega-1}D)$ in Corollary 3.1 improves upon it, since here we have $m\mu\nu > D$.

Finally, a problem similar to Problem 11 is solved in [GR06], except that it is not assumed that $x_1, \ldots, x_\nu$ are distinct. For simple roots and under some genericity assumption on the points $\{(x_i, y_{i,1}, \ldots, y_{i,r})\}_{1 \leqslant i \leqslant \nu}$, this algorithm uses $\mathcal{O}(\nu^{2+1/r})$ operations to compute a polynomial Q which satisfies *(i)*, *(ii)*, and *(iii)* with $\mu = 1$. However, the cost analysis is not clear to us in the general case with multiple roots ($\mu > 1$).

3.1.3 Computing shifted Popov bases of multivariate interpolants

Here we study a slightly more general multivariate interpolation problem, Problem 12 below, motivated in particular by the interpolation step of the soft-decoding of Reed-Solomon codes [KV03a, Section III], where $x_1, \ldots, x_\nu$ are not distinct, and by robustness of Private Information Retrieval [DGH12], where one wants a complete shifted reduced basis of interpolants. The points where Problem 12 generalizes Problem 11 are the following:

- the points $x_1, \ldots, x_\nu$ do not have to be pairwise distinct;
- it supports a more general degree measure for interpolants, via the use of any shifted degree instead of the weighted degree;
- it asks for a whole shifted reduced basis of interpolants, allowing in particular to retrieve interpolants that satisfy degree constraints as explained in Remark 1.9;
- it offers more flexibility on the structure of the multiplicities of the roots, since any monomial set which is stable under division is allowed while Problem 11 only accepts sets defined by some total degree bound.

We will obtain an efficient algorithm thanks to a direct reduction to the problem of computing minimal interpolant bases (Problem 4 with a Jordan multiplication matrix), without considering solution bases (Problem 9). We recall that, while our algorithms for solution bases are efficient only when the number of polynomial equations n is controlled with respect to the number of unknowns m, our cost bounds concerning interpolant bases in Theorems 2.19 and 2.20 do not involve such a restriction.

Problem. As before, we want to find a multivariate polynomial $Q \in \mathbb{K}[X, Y_1, \ldots, Y_r]$ which vanishes at some given points $\{(x_k, y_{k,1}, y_{k,2}, \ldots, y_{k,r}),\ 1 \leqslant k \leqslant \nu\}$. Here, we do not require that the x_k be pairwise distinct, and the points are not associated with integer multiplicities but with prescribed *supports* $\{\boldsymbol{\mu}_k,\ 1 \leqslant k \leqslant \nu\} \subset \mathbb{Z}_{\geqslant 0}^{r+1}$.

In this context, for a point $(x, y_1, \dots, y_r) \in \mathbb{K}^{r+1}$ and a finite exponent set $\boldsymbol{\mu} \subset \mathbb{Z}_{\geqslant 0}^{r+1}$, we say that a polynomial Q *vanishes at* $(x, y_1, \dots, y_r)$ *with support* $\boldsymbol{\mu}$ if the shifted polynomial $Q(X + x, Y_1 + y_1, \dots, Y_r + y_r)$ has no monomial with exponent in $\boldsymbol{\mu}$. When the support has the specific form $\boldsymbol{\mu} = \{(i, j_1, \dots, j_r) \in \mathbb{Z}_{\geqslant 0}^{r+1} \mid i + j_1 + \cdots + j_r < \mu\}$ for some positive integer μ, then this notion of vanishing with support $\boldsymbol{\mu}$ coincides with the definition of having a zero with multiplicity μ given in Section 3.1.1.

Besides, in coding theory applications, the solution $Q(X, Y_1, \dots, Y_r)$ should also have sufficiently small weighted degree $\mathrm{wdeg}_{w_1,\dots,w_r}(Q)$ for some given weights $(w_1, \dots, w_r) \in \mathbb{Z}^r$. Here, we will focus on a more general type of degree minimization.

To make this situation fit in the framework of shifted reduced bases of $\mathbb{K}[X]$-modules, we first require that we are given an exponent set $\mathcal{S} \subseteq \mathbb{Z}_{\geqslant 0}^r$ such that any monomial $X^i Y_1^{j_1} \cdots Y_r^{j_r}$ appearing in a solution Q should satisfy $(j_1, \dots, j_r) \in \mathcal{S}$. As a consequence, we can write $Q = \sum_{(j_1,\dots,j_r)\in\mathcal{S}} p_{j_1,\dots,j_r}(X) Y_1^{j_1} \cdots Y_r^{j_r}$, and having chosen an arbitrary ordering of the elements in $\mathcal{S}$, we can identify Q with $\mathbf{p} = [p_{j_1,\dots,j_r}]_{(j_1,\dots,j_r)\in\mathcal{S}} \in \mathbb{K}[X]^{1\times m}$, where m is the cardinality of $\mathcal{S}$.

Furthermore, since $Q(X, X^{w_1} Y_1, \cdots, X^{w_r} Y_r) = \sum_{(j_1,\dots,j_r)\in\mathcal{S}} X^{w_1 j_1 + \cdots + w_r j_r} p_j(X) Y^j$, by definition the weighted degree $\mathrm{wdeg}_{w_1,\dots,w_r}(Q)$ is precisely the $\mathbf{s}$-degree of $\mathbf{p}$ for the shift $\mathbf{s} = (w_1 j_1 + \cdots + w_r j_r)_{(j_1,\dots,j_r)\in\mathcal{S}}$.

Our goal will be to express the multivariate interpolation problem studied in this section, which we are still defining, as a shifted minimal interpolant basis problem. The link is almost complete, except for an additional assumption to ensure that the set of solutions is a $\mathbb{K}[X]$-module. We require that each considered support $\boldsymbol{\mu}$ satisfy

$$\text{if } (i, j_1, \dots, j_r) \in \boldsymbol{\mu} \text{ and } i > 0, \text{ then } (i-1, j_1, \dots, j_r) \in \boldsymbol{\mu}. \tag{3.1}$$

Then, the set

$$\mathcal{M}_{\mathrm{int}} = \left\{ \mathbf{p} = [p_{j_1,\dots,j_r}]_{(j_1,\dots,j_r)\in\mathcal{S}} \in \mathbb{K}[X]^{1\times m} \;\middle|\; \sum_{(j_1,\dots,j_r)\in\mathcal{S}} p_{j_1,\dots,j_r}(X) Y_1^{j_1} \cdots Y_r^{j_r} \right.$$
$$\left. \text{vanishes at } (x_k, y_{k,1}, \dots, y_{k,r}) \text{ with support } \boldsymbol{\mu}_k \text{ for } 1 \leqslant k \leqslant \nu \right\}$$

is a $\mathbb{K}[X]$-module, as can be seen from the equality

$$(XQ)(X + x, Y_1 + y_1, \dots, Y_r + y_r) = (X + x) Q(X + x, Y + y_1, \dots, Y_r + y_r). \tag{3.2}$$

This leads us to Problem 12.

In particular, since an $\mathbf{s}$-reduced basis of $\mathcal{M}_{\mathrm{int}}$ contains a row whose $\mathbf{s}$-degree is minimal among all elements of $\mathcal{M}_{\mathrm{int}}$, one may compute an interpolant with sufficiently small weighted degree from such a basis for a shift chosen as above. Besides, we note that in some cases it is important to return a whole basis of interpolants and not only a single one of small degree (see for example [DGH12]).

Reduction. We now show that this problem is a relation basis problem with the input module being defined by a dual basis (see Section 2.1.1); as such, and since the corresponding multiplication matrix of X is a Jordan matrix, it can be embedded in the framework

Problem 12 – *Minimal basis of multivariate interpolants*

Input:

- pairwise distinct points $\{(x_k, y_{k,1}, \ldots, y_{k,r}) \in \mathbb{K}^{r+1}, 1 \leqslant k \leqslant \nu\}$,
- supports $\{\boldsymbol{\mu}_k \subset \mathbb{Z}_{\geqslant 0}^{r+1}, 1 \leqslant k \leqslant \nu\}$ which all satisfy Eq. (3.1),
- exponent set $\mathcal{S} \subset \mathbb{Z}_{\geqslant 0}^{r}$ of cardinality m,
- shift $\mathbf{s} \in \mathbb{Z}^m$.

Output:

- a matrix $\mathbf{P} \in \mathbb{K}[X]^{m \times m}$ such that
  - the rows of $\mathbf{P}$ form a basis of the $\mathbb{K}[X]$-module $\mathcal{M}_{\text{int}}$,
  - $\mathbf{P}$ is $\mathbf{s}$-reduced.

of minimal interpolant bases (Section 2.4). We consider the multivariate polynomial ring $\mathbb{K}[X, \boldsymbol{Y}] = \mathbb{K}[X, Y_1, \ldots, Y_r]$ and the $\mathbb{K}$-linear functionals

$$\{\ell_{i,j_1,\ldots,j_r,k} : \mathbb{K}[X, \boldsymbol{Y}] \to \mathbb{K}, \quad (i, j_1, \ldots, j_r) \in \boldsymbol{\mu}_k, 1 \leqslant k \leqslant \nu\},$$

where for all $Q \in \mathbb{K}[X, \boldsymbol{Y}]$, $\ell_{i,j_1,\ldots,j_r,k}(Q)$ is the coefficient of the monomial $X^i Y_1^{j_1} \cdots Y_r^{j_r}$ in the shifted polynomial $Q(X + x_k, Y_1 + y_{k,1}, \ldots, Y_r + y_{k,r})$. These functionals are linearly independent, and the intersection $\mathfrak{K}$ of their kernels is the set of polynomials in $\mathbb{K}[X, \boldsymbol{Y}]$ vanishing at $(x_k, y_{k,1}, \ldots, y_{k,r})$ with support $\boldsymbol{\mu}_k$ for all k. The quotient $\mathbb{K}[X, \boldsymbol{Y}]/\mathfrak{K}$ is a $\mathbb{K}$-vector space of dimension $D = D_1 + \cdots + D_\nu$ where D_k is the cardinality of $\boldsymbol{\mu}_k$; it is thus isomorphic to $\mathbb{K}^D$, with a basis of the dual space being given by the functionals.

Our assumption on the supports implies that $\mathbb{K}[X, \boldsymbol{Y}]/\mathfrak{K}$ is a $\mathbb{K}[X]$-module; we now describe the corresponding multiplication matrix $\mathbf{J} \in \mathbb{K}^{D \times D}$. For a given k, let us order the functionals in such a way that, for any $(i, j_1, \ldots, j_r)$ such that $(i + 1, j_1, \ldots, j_r)$ is in $\boldsymbol{\mu}_k$, the successor of $\ell_{i,j_1,\ldots,j_r,k}$ is $\ell_{i+1,j_1,\ldots,j_r,k}$. Equation (3.2) implies that

$$\ell_{i,j_1,\ldots,j_r,k}(XQ) = \ell_{i-1,j_1,\ldots,j_r,k}(Q) + x_k \ell_{i,j_1,\ldots,j_r,k}(Q) \tag{3.3}$$

holds for all $Q \in \mathbb{K}[X, \boldsymbol{Y}]$, all $k \in \{1, \ldots, \nu\}$, and all $(i, j_1, \ldots, j_r) \in \boldsymbol{\mu}_k$ with $i > 0$.

Hence, $\mathbf{J}$ is block diagonal with diagonal blocks $\mathbf{J}_1, \ldots, \mathbf{J}_\nu$, where $\mathbf{J}_k$ is a $D_k \times D_k$ Jordan matrix with only eigenvalue x_k and block dimensions given by the support $\boldsymbol{\mu}_k$. More precisely, defining

$$\Lambda_k = \{(j_1, \ldots, j_r) \in \mathbb{Z}_{\geqslant 0}^r \mid (i, j_1, \ldots, j_r) \in \boldsymbol{\mu}_k \text{ for some } i\}$$

and

$$D_{k,j_1,\ldots,j_r} = \max\{i \in \mathbb{Z}_{\geqslant 0} \mid (i, j_1, \ldots, j_r) \in \boldsymbol{\mu}_k\},$$

for each $(j_1, \ldots, j_r) \in \Lambda_k$, we have the disjoint union

$$\boldsymbol{\mu}_k = \bigcup_{(j_1,\ldots,j_r) \in \Lambda_k} \{(i, j_1, \ldots, j_r), 0 \leqslant i \leqslant D_{k,j_1,\ldots,j_r}\}.$$

Then, $\mathbf{J}_k$ is block diagonal with $\mathrm{Card}(\Lambda_k)$ blocks: to each $(j_1, \ldots, j_r) \in \Lambda_k$ corresponds a $D_{k,j_1,\ldots,j_r} \times D_{k,j_1,\ldots,j_r}$ Jordan block with eigenvalue x_k. It is reasonable to consider $x_1, \ldots, x_\nu$

ordered as we would like for a standard representation of $\mathbf{J}$. For example, in problems coming from coding theory, these points are part of the code itself, so the reordering can be done as a pre-computation as soon as the code is fixed.

To complete the reduction to Problem 4, it remains to construct the input matrix of evaluations $\mathbf{F} \in \mathbb{K}^{m\times D}$. For each exponent $(\gamma_1, \ldots, \gamma_r) \in \mathcal{S}$ we consider the monomial $Y_1^{\gamma_1} \cdots Y_r^{\gamma_r}$ and take its image in $\mathbb{K}[X, \boldsymbol{Y}]/\mathfrak{K}$: this is the vector $\mathbf{f}_{\gamma_1,\ldots,\gamma_r} \in \mathbb{K}^D$ having for entries the evaluations of the functionals $\ell_{i,j_1,\ldots,j_r,k}$ at $Y_1^{\gamma_1} \cdots Y_r^{\gamma_r}$. Let then $\mathbf{F}$ be the matrix in $\mathbb{K}^{m\times D}$ with rows $(\mathbf{f}_{\gamma_1,\ldots,\gamma_r})_{(\gamma_1,\ldots,\gamma_r)\in\mathcal{S}}$: our construction shows that a row $\mathbf{p} = [p_{\gamma_1,\ldots,\gamma_r}]_{(\gamma_1,\ldots,\gamma_r)\in\mathcal{S}} \in \mathbb{K}[X]^{1\times m}$ is in $\mathcal{M}_{\text{int}}$ if and only if it is an interpolant of $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$.

Result. To make the above reduction to Problem 4 efficient, we make the assumption that the exponent sets $\mathcal{S}$ and $\boldsymbol{\mu}_k$ are *stable under division*. This means that if some exponent $(j_1, \ldots, j_r)$ is in $\mathcal{S}$ then all $(j'_1, \ldots, j'_r)$ such that $j'_1 \leqslant j_1, \ldots, j'_r \leqslant j_r$ also belong to $\mathcal{S}$; and similarly if $(i, j_1, \ldots, j_r)$ is in a support $\boldsymbol{\mu}_k$, then all $(i', j'_1, \ldots, j'_r)$ such that $i' \leqslant i, j'_1 \leqslant j_1, \ldots, j'_r \leqslant j_r$ belong to $\boldsymbol{\mu}_k$.

This assumption is satisfied in all the particular cases we will consider, where $\mathcal{S}$ and the $\boldsymbol{\mu}_k$ are often given through a bound on the total degree of their elements, as in Sections 3.1.1 and 3.1.2. For example, in applications to list-decoding algorithms we typically have $\mathcal{S} = \{(j_1, \ldots, j_r) \in \mathbb{Z}^r_{\geqslant 0} \mid j_1 + \cdots + j_r \leqslant \lambda\}$ for some positive integer λ; in this context, we thus have $m = \binom{r+\lambda}{r}$, and the parameter $D = \mathrm{Card}(\boldsymbol{\mu}_1) + \cdots + \mathrm{Card}(\boldsymbol{\mu}_\nu)$ is sometimes called the *cost* [KV03a, Section III]. As a side note, we also remark that this assumption also implies that $\mathfrak{K}$ is a zero-dimensional ideal of $\mathbb{K}[X, \boldsymbol{Y}]$.

Most importantly, using the straightforward extension of Eq. (3.3) to multiplication by $Y_1, \ldots, Y_r$, it allows us to compute all entries $\ell_{i,j_1,\ldots,j_r,k}(Y_1^{\gamma_1} \cdots Y_r^{\gamma_r})$ of the matrix $\mathbf{F}$ inductively in $\mathcal{O}(mD)$ operations, which is negligible compared to the cost of solving the resulting instance of Problem 4.

We then get the following result as a corollary of Theorems 2.19 and 2.20. We refer to Theorem 2.12 concerning the case where $D = \mathrm{Card}(\boldsymbol{\mu}_1) + \cdots + \mathrm{Card}(\boldsymbol{\mu}_\nu)$ is in $\mathcal{O}(m)$, where $m = \mathrm{Card}(\mathcal{S})$.

Corollary 3.2. *Let D be the sum of the cardinalities of the multiplicity supports, let m be the cardinality of $\mathcal{S}$, and assume that $\mathcal{S}$ and $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_\nu$ are stable under division. There is a deterministic algorithm which solves Problem 12 using*

$$\mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(D)\log(D/m) + m^{\omega-1}\mathsf{M}(\xi)\log(\xi/m))$$

operations in $\mathbb{K}$, where $\xi = |\mathbf{s} - \min(\mathbf{s})|$. Furthermore, for an arbitrary shift $\mathbf{s}$, there is a deterministic algorithm which solves Problem 12 using

$$\mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(D)\log(D/m)^2)$$

operations in $\mathbb{K}$ and which returns the $\mathbf{s}$-Popov basis of $\mathcal{M}_{\text{int}}$.

The term $\mathcal{O}(m^{\omega-1}\mathsf{M}(\xi)\log(\xi/m))$ in the first cost bound can be neglected if $\xi \in \mathcal{O}(D)$; this is for example the case in the contexts of bivariate interpolation for soft- or list-decoding of Reed-Solomon codes, as detailed below in Section 3.1.4. Thus, in these contexts, the logarithmic factor in the cost is in $\mathcal{O}(\log(D)^2)$, as we claimed in Table 3.1.

However, we do not have such a bound on ξ in the list-decoding of Parvaresh-Vardy codes and folded Reed-Solomon codes. Thus, in these cases we rely on the second algorithm, and the logarithmic factor in the cost bound is $\log(D)^3$. To the best of our knowledge, this is the best known cost bound for a deterministic algorithm for these list-decoding problems, improving upon [Bus08, Bra10, CH12]. For this particular case, a slightly faster, but probabilistic, algorithm was presented in Section 3.1.1, with cost bound $\mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(D))$ (see Corollary 3.1).

The algorithm of Section 3.1.1 does not cover the same generality, as summarized at the beginning of this section. In particular, it requires $x_1, \ldots, x_\nu$ to be distinct and it returns a single interpolant which satisfies prescribed degree constraints. It is sometimes important, for example in the use of list-decoding algorithms for Private Information Retrieval [DGH12], to return the whole basis of interpolants. To the best of our knowledge, our algorithm is the first one which costs $\mathcal{O}\tilde{}(m^{\omega-1}D)$ for this task; this improves in particular on the approach of [Bus08, Bra10, CH12] based on row reduction, which we presented in Section 3.1.2. There are also interesting situations where $x_1, \ldots, x_\nu$ are not distinct, as we detail now in the next section.

3.1.4 Soft-decoding of Reed-Solomon codes

Kötter and Vardy developed an extension of the Guruswami-Sudan algorithm to the context of soft-decoding of Reed-Solomon codes, where one has reliability information on the symbols of the received word. This algorithm involves the *soft-interpolation step* [KV03a, Section III] which deals with Problem 12 with $r = 1$ and $\mathcal{S} = \{0, \ldots, m-1\}$; here, the points $x_1, \ldots, x_\nu$ are not necessarily pairwise distinct, and to each x_k for $1 \leqslant k \leqslant \nu$ is associated a multiplicity parameter μ_k and a corresponding support $\boldsymbol{\mu}_k = \{(i,j) \mid i+j < \mu_k\}$. In [KV03a], the quantity $D = \sum_{1 \leqslant k \leqslant \nu} \binom{\mu_k+1}{2}$ is called the *cost*; we recall that it corresponds to the number of linear equations over $\mathbb{K}$ that one obtains after linearizing the condition of belonging to the interpolant module $\mathcal{M}_{\text{int}}$.

As explained above, here the shift $\mathbf{s}$ takes the form $\mathbf{s} = (0, w, 2w, \ldots, (m-1)w)$, with $w+1$ being the message length of the considered Reed-Solomon code. In this context, one chooses for m the smallest integer such that the number of linear unknowns in the linearized problem is more than D. Because this number of unknowns is directly linked to $|\mathbf{s}|$, this leads to the bound $|\mathbf{s}| \in \mathcal{O}(D)$, which can be proven for example using [KV03a, Lemma 1 and Equations (10) and (11)]. Thus, from Corollary 3.2, our algorithm solves the soft-interpolation step using $\mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(D)\log(D/m))$ operations in $\mathbb{K}$. To the best of our knowledge, this is the best known cost bound to solve this problem.

The iterative algorithms in [Bec92, VBB92, Köt96, NH00, McE03], which we mentioned in Section 3.1.2 concerning the Guruswami-Sudan interpolation step, also work in this more general situation. They use $\mathcal{O}(mD^2)$ operations in $\mathbb{K}$, since the considered input shift satisfies $|\mathbf{s} - \min(\mathbf{s})| \in \mathcal{O}(D)$. Based on the same recurrence relations, our fast Algorithm 28 can be seen as a divide-and-conquer version of these algorithms.

A previous divide-and-conquer algorithm can be found in [Nie14], focusing on the case of identical multiplicities $\mu = \mu_1 = \cdots = \mu_\nu$. The recursion is on the number of points ν, and using fast multiplication of the bases obtained recursively, this algorithm has cost bound $\mathcal{O}(m^2 D\mu) + \mathcal{O}\tilde{}(m^\omega D/\mu)$ [Nie14, Proposition 3]. Both terms in this bound

are larger than the cost $\tilde{\mathcal{O}}(m^{\omega-1}D)$ obtained here; we note that in [Nie14], the bases computed recursively may have size as large as $\Theta(m^2D/\mu)$, with $\mu < m$, while in our algorithm their size is always bounded by $\mathcal{O}(mD)$ (in this case with $|\mathbf{s}-\min(\mathbf{s})| \in \mathcal{O}(D)$). While the algorithm of [Nie14] can be adapted to the case of distinct multiplicities, it is not clear to us what cost bound this would lead to.

The approach based on row reduction of a polynomial matrix, mentioned in Section 3.1.2, was also developed for this more general interpolation problem in [Ale02, Ale05, LO06]. It consists in first building a basis $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$ of the $\mathbb{K}[X]$-module of interpolants $\mathcal{M}_{\text{int}}$, and in then reducing this basis for the given shift $\mathbf{s}$ to obtain a $\mathbf{s}$-minimal basis of interpolants. The maximal degree in $\mathbf{A}$ is

$$\beta = \sum_{1\leqslant k\leqslant \nu} \frac{\max\{\mu_i \mid 1 \leqslant i \leqslant \nu \text{ and } x_i = x_k\}}{\text{Card}(\{1 \leqslant i \leqslant \nu \mid x_i = x_k\})};$$

one can check using $\mu_k < m$ for all k that

$$D = \sum_{1\leqslant k\leqslant \nu} \frac{\mu_k(\mu_k+1)}{2} \leqslant \frac{m}{2} \sum_{1\leqslant k\leqslant \nu} \mu_k \leqslant \frac{m\beta}{2}.$$

Using the fast deterministic reduction algorithm in [GSSV12], this approach has cost bound $\mathcal{O}(m^\omega \mathsf{M}(\beta)(\log(m)^2 + \log(\beta)))$; the cost bound of our algorithm is thus smaller by a factor $\tilde{\mathcal{O}}(m\beta/D)$.

In [Zeh13, Section 5.1] the so-called key equations commonly used in the decoding of Reed-Solomon codes were generalized to this soft-interpolation step. We detail in Section 11.2.4 how to adapt our results based on fast structured linear algebra (Theorem 2.25 and Corollary 3.1), in order to solve these equations efficiently. In this approach, the set of points $\{(x_k, y_k), 1 \leqslant k \leqslant \nu\}$ is partitioned as $\mathcal{P}_1 \cup \cdots \cup \mathcal{P}_q$, where in each set $\mathcal{P}_h$ the points have pairwise distinct x-coordinates. We further write $\mu^{(h)} = \max\{\mu_k \mid 1 \leqslant i \leqslant \nu \text{ and } (x_k, y_k) \in \mathcal{P}_h\}$ for each h, and $n = \sum_{1\leqslant h\leqslant q} \mu^{(h)}$. Then, the cost bound of this approach is $\mathcal{O}((m+n)^{\omega-1}\mathsf{M}(D)\log(D))$, with a probabilistic algorithm. We note that n may be significantly larger than m. Our algorithm supporting Corollary 3.2 is deterministic and has a better cost.

All the mentioned algorithms for the interpolation step of list- or soft-decoding of Reed-Solomon codes, including the ones presented in this thesis, can be used in conjunction with the *re-encoding technique* [WB86, KV03b] for the decoding problem (see for example Section 11.2.2).

3.1.5 General Coppersmith technique over $\mathbb{K}[X]$

The *Coppersmith technique* [Hås86, GTV90, Cop96, HG01] primarily refers to a lattice-based method to compute small modular roots of a polynomial $F(Y)$ over the integers $\mathbb{Z}$. It has been largely studied in the last twenty years, in particular for its applications in cryptography, and extends to polynomials over several other domains [CH15].

Here, we focus on the case of a polynomial $F(Y)$ over the univariate polynomials $\mathbb{K}[X]$. We will see that the list-decoding problem for Reed-Solomon codes [GS99] precisely asks

to compute a small modular root, and therefore that the Guruswami-Sudan algorithm can be interpreted as an instance of the Coppersmith technique [Ber11, CH11].

We will present the Coppersmith technique over $\mathbb{K}[X]$ and then focus on its first step, called the *interpolation step*. We show that this step can be efficiently reduced to a system of linear modular equations over $\mathbb{K}[X]$; solving the latter with the algorithms presented above leads to an improvement upon previous algorithms, which rely on fast row reduction. The problem reduction we give can be seen as a generalization of the derivation of extended key equations in the reduction of the Guruswami-Sudan interpolation step to a system of modular equations [RR00, ZGA11] (see also Section 11.1).

In what follows, $\mathbb{K}[X][Y]$ stands for the set of univariate polynomials in Y over $\mathbb{K}[X]$. While this is essentially the set of bivariate polynomials in X and Y, in the context here we will most often see its elements as univariate polynomials.

Small modular roots. In its version over $\mathbb{K}[X]$, the Coppersmith technique solves Problem 13.

Problem 13 – *Small modular roots*

Input:

- positive integers d and ν,
- nonnegative integers w and t,
- $M \in \mathbb{K}[X]$ of degree ν,
- $F \in \mathbb{K}[X][Y]$ monic of degree d with coefficients of degree $< \nu$.

Output:

- all polynomials $p \in \mathbb{K}[X]$ such that
  - $\deg(\gcd(F(p), M)) \geqslant t$;
  - $\deg(p) \leqslant w$.

We note that one may consider M monic without loss of generality. The size of the input, in terms of the number of field elements used to represent it, is at most $d\nu$ for F and ν for M. The size of the output is not obvious to analyze: the number of solutions depends on the four parameters d, ν, w and t, and may be exponential in d or ν.

Under the assumption $t^2 > w\nu d$, the $\mathbb{K}[X]$ version of the Coppersmith technique solves Problem 13 in time polynomial in d and ν, with two main steps:

- the *interpolation step* which builds a polynomial $Q \in \mathbb{K}[X][Y]$ that satisfies $Q(p) = 0$ for all p that are solutions to Problem 13;
- the *root-finding step* which computes the roots of $Q(Y)$ in $\mathbb{K}[X]$ and returns those that are actual solutions to Problem 13.

For more details, we refer to the discussion after Problem 14, and to [CH11, CH15].

The decoding of Reed-Solomon codes can be seen as a particular case of this problem. In that context, $w + 1$ is the message length (or dimension of the code) and $\nu - t$ is an

upper bound on the number of errors that can be corrected by the decoder. We have as input the code evaluation points $x_1, \ldots, x_\nu$, which are pairwise distinct in $\mathbb{K}$, as well as a received word $(y_1, \ldots, y_\nu) \in \mathbb{K}^\nu$. Then, the goal is to find all $p \in \mathbb{K}[X]$ such that $\deg(p) \leqslant w$ and $p(x_i) = y_i$ for at least t values of $i \in \{1, \ldots, \nu\}$; the number of such i's is the number of correct locations, called the agreement.

Let us define L as the Lagrange polynomial such that $L(x_i) = y_i$ for all i, as well as $M = (X - x_1) \cdots (X - x_\nu)$ and $F = Y - L$. Then, by construction $\deg(\gcd(F(p), M))$ counts the number of indices i for which $p(x_i) = y_i$: it measures the agreement, for a given $p \in \mathbb{K}[X]$. As a result, for this particular input M and F, Problem 13 precisely corresponds to the decoding problem. Depending on the parameters, it can be solved by unique decoders such as the Welch-Berlekamp algorithm [WB86], or by list-decoders such as the Guruswami-Sudan algorithm [GS99], or it can have an exponential number of solutions (only if $t^2 \leqslant w\nu$).

Remark 3.3. Most previous works on the latter algorithm focus on reducing the cost of the interpolation step; we refer to Section 3.1.2 for an overview. Algorithms for the root-finding step have been discussed for example in [RR00, Ale05, Rot07, Ber11]. While this second step has been considered until now as negligible in terms of efficiency, recent progress on the first step is closing the gap between the two, at least from a theoretical point of view. Therefore, it seems that a proper study should be conducted with regards to possible improvements on the root-finding step. ☕

The interpolation step. In this document, we focus on the interpolation step, which can be formalized as follows (Problem 14).

Problem 14 – *Interpolation step*

Input:

- positive integers d and ν,
- nonnegative integers w and t,
- $M \in \mathbb{K}[X]$ of degree ν,
- $F \in \mathbb{K}[X][Y]$ monic of degree d with coefficients of degree $< \nu$.
- positive integer μ.

Output:

- a nonzero polynomial $Q \in \mathbb{K}[X][Y]$ such that
  - Q belongs to the ideal
  $$\mathcal{I} = \langle M, F \rangle^\mu = \langle M^\mu, M^{\mu-1}F, \ldots, MF^{\mu-1}, F^\mu \rangle;$$
  - $\deg_X(Q(X^w Y)) < \mu t$.

Writing $Q = Q_0 + Q_1 Y + Q_2 Y^2 + \cdots$, the second item requires that $\deg(Q_j) < \mu t - jw$ for all $j \leqslant \deg_Y(Q)$. One may note that this corresponds to requiring that, seeing Q as a bivariate polynomial, its w-weighted degree be less than μt. In particular, $\deg_Y(Q)$ cannot grow arbitrarily large: it is bounded from above by $\mu t / w$.

As in list-decoding algorithms, the integer μ is called the multiplicity parameter and is introduced to make the technique work in more cases. For example, the Sudan algorithm [Sud97] is with $\mu = 1$ and is able to correct up to about $\nu - \sqrt{2w\nu}$ errors, while the Guruswami-Sudan algorithm [GS99] improves this bound to $\nu - \sqrt{w\nu}$ by using $\mu > 1$.

To briefly justify this interpolation step, suppose that we have computed a solution Q to Problem 14. Then, every solution p to Problem 13 satisfies

- $\deg(\gcd(F(p), M)) \geqslant t$ and thus $\deg(\gcd(Q(p), M^\mu)) \geqslant \mu t$ since Q belongs to $\mathcal{I}$,
- $\deg(p) \leqslant w$ and thus $\deg(Q(p)) \leqslant \deg_X(Q(X^w Y)) < \mu t$,

which together imply $Q(p) = 0$. Therefore, having Q, to solve Problem 13, it remains to compute its roots over $\mathbb{K}[X]$ and to verify which of them are solutions to Problem 13.

The choice of the parameter $\mu \geqslant 1$ can be done by counting how many linear unknowns and linear equations we have in the $\mathbb{K}$-linear system corresponding to the given instance of Problem 14, and by taking μ sufficiently large so that there are more unknowns than equations. This is only feasible under some assumption on the parameters ν, d, w, t.

Namely, assuming that $t^2 > w\nu d$, one may choose

$$\mu \;=\; \left\lfloor \frac{w(\nu d - t)}{t^2 - w\nu d} \right\rfloor + 1 \quad \text{and} \quad \lambda = \left\lfloor \mu \frac{t}{w} \right\rfloor , \tag{3.4}$$

which are such that there is a solution Q to Problem 14 with $\deg_Y(Q) \leqslant \lambda$. Indeed, this choice implies the inequality

$$\sum_{j \leqslant \lambda} (\mu t - jw) \;\; > \;\; \sum_{i < \mu} (\mu - i)\nu d \, , \tag{3.5}$$

which precisely states that there are more unknowns than equations in the linearized problem. (We will also derive this inequality from the reduction-based approach in Section 10.1.) We remark that λ is called the *list-size parameter*, since it also bounds the number of solutions to Problem 13. We deduce that this number is at most $\lambda < 2t\nu d$.

Result. In the case of Reed-Solomon list-decoding, in Section 3.1.2 we mentioned algorithms based on a linearization into a structured system over $\mathbb{K}$, others using a top-down approach relying on row reduction, and finally those following a bottom-up, incremental procedure. Some of these algorithms exploit the known roots of M, such as the bottom-up ones and the one in [OS99] which involves block-Vandermonde matrices. In general, these roots are unknown and may not be in $\mathbb{K}$.

On the other hand, the top-down approach can be seen as the polynomial counterpart of the CRT list-decoding in [Bon02], as stated in [Rei03], and directly generalizes to any inputs M and F. As such, it is the analogue of the general Coppersmith technique over the integers; we present it in more detail in Section 10.1, following [CH15]. This solves Problem 14 using $\mathcal{O}\tilde{}(\lambda^\omega \mu\nu)$ operations in $\mathbb{K}$.

Then, we extend our improvement of the interpolation step in the Guruswami-Sudan algorithm (Section 3.1.2) to the more general context here. We show how the first requirement on Q in Problem 14, that it belongs to $\mathcal{I}$, can be efficiently rewritten as a system

of linear modular equations over $\mathbb{K}[X]$ (Section 10.2). Using our fast algorithms for such systems, this allows us to solve Problem 14 using $\tilde{\mathcal{O}}(\lambda^{\omega-1}\mu^2\nu d)$ field operations. This is a speed-up factor of $\lambda/(\mu d)$ with respect to the previously fastest known algorithm; we have $\lambda \geqslant \mu d$ by choice of the parameters.

Compared to the generalization of the extended key equation of [RR00, ZGA11] to several variables and more general constraints (Sections 3.1.2 and 11.1), which was based on Hasse derivatives, here we perform the reduction to a system of equations without requiring the knowledge of roots and multiplicities of the modulus M. While we present it only in the case of one variable Y, we expect that this extends to several variables.

Remark 3.4. Multivariate extensions of Problem 13 arise in list-decoding problems, namely for Parvaresh-Vardy codes and folded Reed-Solomon codes [PV05, GR08], and in questions related to robustness in Private Information Retrieval [DGH12]. With several variables, the root-finding step becomes more involved. Besides, in [DGH12] the interpolation step should output several independent Q in order to be able to find the roots; a single one is sufficient in list-decoding contexts because there are others that we know by construction.

Another issue in multivariate extensions of Problem 13 is the choice of parameters: in this case as well, it is made so that some well-identified inequality is satisfied, yet we do not have a nice closed-form expression, to the best of our knowledge. ☕

3.2 Computing shifted Popov forms of polynomial matrices

In this section, we present our results concerning the computation of shifted Popov forms of polynomial matrices. We give three main contributions. First, we adapt a technique from [GSSV12] to give a reduction of the degrees in the input matrix which behaves well regarding shifted Popov form computation; while it only slightly increases the dimensions, it also ensures that the degree is at most the average column and row degrees of the original matrix. Second, we give a fast, Las Vegas randomized algorithm to compute shifted Popov forms for an arbitrary shift, relying on Smith form computations [Sto03, GS11, Gup11] and on our result on fast shifted Popov solution basis from Section 2.5. Finally, we design a similarly fast, yet deterministic algorithm for the particular case of Hermite form computation. The latter algorithm starts by computing the diagonal entries of the Hermite form, which may be modified to deterministically yield the determinant of a polynomial matrix.

3.2.1 Overview

Our problem asks to find canonical forms of polynomial matrices: given a polynomial matrix $\mathbf{A}$ and a shift $\mathbf{s}$, we want to compute the $\mathbf{s}$-Popov form of $\mathbf{A}$. This situation differs from those studied above in this document, for the following reason. In the computation of relation bases and its variants presented in Chapter 2, the module of which we wanted to compute a shifted Popov basis was given to us implicitly, essentially via a set of equations.

Here, we are interested in the $\mathbb{K}[X]$-module $\mathcal{M}$ generated by the rows of the input matrix $\mathbf{A}$, also called the row space of $\mathbf{A}$: it is is described explicitly, by a generating set.

As in most previous work on fast algorithms for this kind of question, we will focus on computing shifted Popov forms of *square nonsingular matrices*. The general case of an input matrix which is rectangular or does not necessarily have full rank has been studied in [BLV06]; a fast solution would require further developments. In terms of modules, assuming that $\mathbf{A}$ has full rank means that the rows of $\mathbf{A}$ form a basis of its row space $\mathcal{M}$, and we want to unimodularly transform this basis of $\mathcal{M}$ into the one in $\mathbf{s}$-Popov form. At the same time, assuming that $\mathbf{A}$ is square means that we focus on the case of a module $\mathcal{M}$ of finite codimension $\deg(\det(\mathbf{A}))$. Being in this situation will in particular help us to better control the degrees in the matrices that we manipulate during the computation.

Problem 15 – *Shifted Popov form*

Input:

- a nonsingular matrix $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$,
- a shift $\mathbf{s} \in \mathbb{Z}^m$.

Output:

- the $\mathbf{s}$-Popov form of $\mathbf{A}$.

Two particularly interesting specific cases of Problem 15 are the computation of the Popov form [Pop72, Kai80] for the *uniform* shift $\mathbf{s} = \mathbf{0}$, and of the Hermite form [Her51, Kai80] for the shift $\mathbf{h} = (0, \delta, 2\delta, \ldots, (m-1)\delta) \in \mathbb{Z}^m_{\geqslant 0}$ where $\delta = m \deg(\mathbf{A})$. As we have discussed in Chapter 1, in addition to some normalization property, the Popov form essentially asks that the row degrees be minimal, while the Hermite form asks that the matrix be triangular. For a broader perspective on the computation of shifted reduced forms, we refer the reader to [BLV99, BLV06].

Problem 16 – *Hermite form*

Input:

- a nonsingular matrix $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$.

Output:

- the Hermite form of $\mathbf{A}$.

During the past decade, there has been a goal to design algorithms that perform various $\mathbb{K}[X]$-linear algebra operations in about the time that it takes to multiply two polynomial matrices having the same dimensions and degree as the input matrix, namely at a cost $\tilde{\mathcal{O}}(m^\omega d)$. *Probabilistic* algorithms with such a cost already exist for a number of polynomial matrix problems, for example for linear system solving [Sto03], Smith normal form computation [Sto03], row reduction [GJV03], and small kernel bases computation [SV05]. Concerning polynomial matrix inversion, the algorithm in [JV05] costs $\tilde{\mathcal{O}}(m^3 d)$, which is quasi-linear in the number of field elements used to represent the inverse. Re-

cently, *deterministic* fast algorithms have been given for linear system solving [GSSV12], minimal kernel bases [ZLS12], and matrix inversion [ZLS15].

For Hermite form computation, Gupta and Storjohann [GS11] gave a Las Vegas randomized algorithm with expected cost $\tilde{\mathcal{O}}(m^3 d)$, later improved to $\tilde{\mathcal{O}}(m^\omega d)$ in [Gup11]. Their algorithm was the first for this task to be both softly cubic in m and softly linear in d. Furthermore, a deterministic algorithm in $\tilde{\mathcal{O}}(m^\omega d)$ has been given for the computation of $\mathbf{0}$-reduced forms and the normalization into $\mathbf{0}$-Popov form [GSSV12, SS11].

In this document, to the best of our knowledge, we give the first algorithm with cost bound $\tilde{\mathcal{O}}(m^\omega d)$ to solve Problem 15 for an arbitrary input shift; this algorithm is presented in Section 3.2.2 and is probabilistic. Furthermore, we design a deterministic algorithm in $\tilde{\mathcal{O}}(m^\omega d)$ for the specific case of Hermite form computation, by exploiting its triangular shape; this is presented in Section 3.2.3.

Yet, in some cases, the cost bound $\tilde{\mathcal{O}}(m^\omega d)$ may be unsatisfactory, namely if the degree d is significantly larger than the average degree of the entries of the matrix $\mathbf{A}$. Recently, it has been a goal to obtain cost bounds that take into account some types of average degrees of the matrices rather than their maximum degree. This has already been achieved for the computation of order bases [Sto06, ZL12], kernel bases [ZLS12], and matrix inversion [ZLS15]. We also achieve this here for the computation of relation bases, since our results summarized in Chapter 2 involve the average column degree of the output in the cost bounds.

Here, we obtain a similar improvement for the computation of shifted Popov forms of a matrix. For this, we make use of the partial linearization framework in [GSSV12, Section 6] and the related *generic determinant bound* $D_{\mathbf{A}}$ for a matrix $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$; we introduce this quantity in our cost bounds. Details about $D_{\mathbf{A}}$ are given in Section 15.1, where we also give more insight into our interest in expressing our cost bounds with $D_{\mathbf{A}}$, rather than $\deg(\mathbf{A})$ for example. This can be summarized into the fact that $D_{\mathbf{A}}$ better reflects the distribution of the degrees in $\mathbf{A}$ than $\deg(\mathbf{A})$; in particular, $D_{\mathbf{A}}/m$ is upper bounded by both the average row degree and the average column degree of $\mathbf{A}$.

We will prove the following statement, which shows that for any instance of Problem 15 or Problem 16 one may always reduce the degrees in the input matrix so that it is not much larger than $D_{\mathbf{A}}/m$. Furthermore, this is done with only a moderate increase of the dimensions of the matrix.

Theorem 3.5. *Let $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$ be nonsingular, and let $\mathbf{s} \in \mathbb{Z}^m$. Using no operation in $\mathbb{K}$, one can build a matrix $\hat{\mathbf{A}} \in \mathbb{K}[X]^{\hat{m}\times\hat{m}}$ and a shift $\hat{\mathbf{s}} \in \mathbb{Z}^{\hat{m}}$ such that*

(i) $m \leqslant \hat{m} < 4m$ and $\deg(\hat{\mathbf{A}}) < 3(1 + D_{\mathbf{A}}/m)$,

(ii) the $\mathbf{s}$-Popov form of $\mathbf{A}$ is the leading principal submatrix of the $\hat{\mathbf{s}}$-Popov form of $\hat{\mathbf{A}}$,

(iii) the Hermite form of $\mathbf{A}$ is the leading principal submatrix of the Hermite form of $\hat{\mathbf{A}}$.

We give a summary of the cost bounds of known fast algorithms for the computation of shifted Popov forms in Table 3.2.

Remark 3.6. From Theorem 3.5, we deduce that any algorithm which computes the Hermite form of $\mathbf{A}$ in $\tilde{\mathcal{O}}(m^\omega \deg(\mathbf{A}))$ operations can be transformed into an algorithm which computes the Hermite form of $\mathbf{A}$ in $\tilde{\mathcal{O}}(m^\omega \lceil D_{\mathbf{A}}/m \rceil)$ operations.

However, the same remark does not hold for an algorithm that would focus on the uniform shift $\mathbf{s} = \mathbf{0}$. Indeed, the corresponding transformed shift $\hat{\mathbf{s}}$ is not uniform anymore, and will actually have amplitude $\max(\hat{\mathbf{s}}) - \min(\hat{\mathbf{s}})$ at least $\deg(\mathbf{A})$. In other words, with this technique to take average degrees into account, computing a $\mathbf{0}$-Popov form reduces to computing a shifted Popov form for a non-uniform shift. This is a further motivation for studying the computation of shifted Popov forms for arbitrary shifts.

In particular, although a deterministic $\tilde{\mathcal{O}}(m^\omega \deg(\mathbf{A}))$ algorithm for the $\mathbf{0}$-Popov form is known [GSSV12, SS11], the degree reduction in Theorem 3.5 does not imply that we can compute the $\mathbf{0}$-Popov form *deterministically* in $\tilde{\mathcal{O}}(m^\omega \lceil D_{\mathbf{A}}/m \rceil)$ operations; to the best of our knowledge, how to do this is still an open question. ☕

Table 3.2: Fast algorithms for shifted reduced forms and shifted Popov forms of a polynomial matrix ($d = \deg(\mathbf{A})$; $\star$ = probabilistic; $\mathrm{amp}(\mathbf{s}) = \max(\mathbf{s}) - \min(\mathbf{s})$).

Algorithm	Problem	Cost bound	
Hafner-McCurley [HM91]	Hermite form	$\tilde{\mathcal{O}}(m^4 d)$	
Storjohann-Labahn [SL96]	Hermite form	$\tilde{\mathcal{O}}(m^{\omega+1} d)$	
Villard [Vil96]	Popov & Hermite forms	$\tilde{\mathcal{O}}(m^{\omega+1} d + (md)^\omega)$	
Alekhnovich [Ale02, Ale05]	weak Popov form	$\tilde{\mathcal{O}}(m^{\omega+1} d)$	
Mulders-Storjohann [MS03b]	Popov & Hermite forms	$\mathcal{O}(m^3 d^2)$	
Giorgi et al. [GJV03]	$\mathbf{0}$-reduction	$\tilde{\mathcal{O}}(m^\omega d)$	$\star$
Sarkar-Storjohann [SS11]	Popov form of $\mathbf{0}$-reduced	$\tilde{\mathcal{O}}(m^\omega d)$	
Gupta-Storjohann [GS11]	Hermite form	$\tilde{\mathcal{O}}(m^\omega d)$	$\star$
Gupta et al. [GSSV12]	$\mathbf{0}$-reduction	$\tilde{\mathcal{O}}(m^\omega d)$	
Folklore, using [GSSV12, SS11]	$\mathbf{s}$-Popov form for any $\mathbf{s}$	$\tilde{\mathcal{O}}(m^\omega(d + \mathrm{amp}(\mathbf{s})))$	
Zhou-Labahn [Zho12, ZL16]	Hermite form	$\tilde{\mathcal{O}}(m^\omega d)$	
Theorem 3.7 and Chapter 15	$\mathbf{s}$-Popov form for any $\mathbf{s}$	$\tilde{\mathcal{O}}(m^\omega \lceil D_{\mathbf{A}}/m \rceil)$	$\star$
Theorem 3.9 and Chapter 16	Hermite form for any $\mathbf{s}$	$\tilde{\mathcal{O}}(m^\omega \lceil D_{\mathbf{A}}/m \rceil)$	

Let us give some more details about the case of an arbitrary shift $\mathbf{s}$. First, in [BLV06], shifted Popov forms are computed for arbitrary shifts, via the computation of kernel bases and relying in particular on [BL00, Algorithm FFFG]; the number of operations is, depending on $\mathbf{s}$, at least quintic in m and quadratic in $\deg(\mathbf{A})$. However we note that comparing this cost to ours would be unfair, since one of the focuses in [BLV06] is to obtain a fraction-free approach, and furthermore the algorithm also returns a transformation from $\mathbf{A}$ to its normal form.

Second, for an arbitrary shift $\mathbf{s}$, there is a folklore reduction to the uniform case, which is based on the fact that a matrix $\mathbf{Q}$ is in $\mathbf{s}$-Popov form if and only if $\mathbf{Q}\mathbf{X}^{\mathbf{s}-\min(\mathbf{s})}$ is in $\mathbf{0}$-Popov form. Then, given $\mathbf{A}$, one would first compute the $\mathbf{0}$-Popov form $\mathbf{P}$ of $\mathbf{A}\mathbf{X}^{\mathbf{s}-\min(\mathbf{s})}$ using [GSSV12, SS11] and then return $\mathbf{P}\mathbf{X}^{\min(\mathbf{s})-\mathbf{s}}$. This technique is used for example in the list-decoding of Reed-Solomon codes via the reduction of a $\mathbb{K}[X]$-module basis, as in [Ale05, Nie13, CH15].

Yet, this approach is not efficient when the amplitude $\mathrm{amp}(\mathbf{s}) = \max(\mathbf{s}) - \min(\mathbf{s})$ of the shift is large, since the reduction leads to compute a $\mathbf{0}$-reduced form of a matrix $\mathbf{A}\mathbf{X}^{\mathbf{s}-\min(\mathbf{s})}$ with large degrees. In general, this solves Problem 15 in $\tilde{\mathcal{O}}(m^\omega(d+\mathrm{amp}(\mathbf{s})))$ field operations; for example, this cost is $\tilde{\mathcal{O}}(m^{\omega+2}d)$ if one wants to compute the Hermite form of $\mathbf{A}$ as in Problem 16, thus choosing the shift $\mathbf{h}$ given above. This is in fact a worst-case situation: as we have seen in Section 1.2.2, one can assume without loss of generality that $\mathrm{amp}(\mathbf{s}) \in \mathcal{O}(m\deg(\det(\mathbf{A}))) \subseteq \mathcal{O}(m^2 d)$.

3.2.2 Computing shifted Popov forms for arbitrary shifts

We obtain the following result, which is a consequence of Proposition 15.9 and the corresponding Algorithm 36.

Theorem 3.7. *Problem 15 can be solved by a Las Vegas randomized algorithm which uses an expected number of*

$$\tilde{\mathcal{O}}(m^\omega \lceil D_{\mathbf{A}}/m \rceil) \subseteq \tilde{\mathcal{O}}(m^\omega \deg(\mathbf{A}))$$

operations in $\mathbb{K}$, assuming that the cardinality of $\mathbb{K}$ is at least $8(4m)^3(3+3D_{\mathbf{A}}/m)$ for the left-hand cost bound, and at least $8m^3\deg(\mathbf{A})$ for the right-hand cost bound.

In this cost bound, the ceiling function indicates that the cost is $\tilde{\mathcal{O}}(m^\omega)$ when $D_{\mathbf{A}}$ is small compared to m. We note that, in this case $D_{\mathbf{A}} \in \mathcal{O}(m)$, the matrix $\mathbf{A}$ has mostly constant entries, and usually, one is rather interested in situations where $m \in \mathcal{O}(D_{\mathbf{A}})$. Then, the cost bound above may be written $\tilde{\mathcal{O}}(m^{\omega-1}D_{\mathbf{A}})$; in terms of average row or column degrees, this is both in $\tilde{\mathcal{O}}(m^{\omega-1}|\mathrm{rdeg}(\mathbf{A})|)$ and in $\tilde{\mathcal{O}}(m^{\omega-1}|\mathrm{cdeg}(\mathbf{A})|)$.

The cost bound that we obtain here is, to the best of our knowledge, the best known cost bound for an arbitrary shift $\mathbf{s}$. Compared to the folklore solution mentioned above, it removes the dependency in $\mathrm{amp}(\mathbf{s})$, which means in some cases a cost bound smaller by a factor m^2. Besides, computing shifted forms for arbitrary shifts also allows us to obtain an improvement for the specific case of the uniform shift when $\mathbf{A}$ has unbalanced degrees. Indeed, to the best of our knowledge, no previously known algorithm computes the $\mathbf{0}$-Popov form of $\mathbf{A}$ using $\tilde{\mathcal{O}}(m^{\omega-1}D_{\mathbf{A}})$ operations, or within a similar bound taking into account the average row or column degree. Interestingly, as mentioned in Remark 3.6, we achieve this here by relying on the computation of a *shifted* Popov form of some transformation of $\mathbf{A}$.

One of the main difficulties in row reduction algorithms is to control the size of the manipulated matrices, that is, the number of coefficients from $\mathbb{K}$ needed for their dense representation. A major issue when dealing with arbitrary shifts is that the size of an $\mathbf{s}$-reduced form of $\mathbf{A}$ may be beyond our target cost, as explained in Section 1.2.2. This gives a further motivation for focusing on the computation of the $\mathbf{s}$-Popov form of $\mathbf{A}$, besides the fact that it is canonical: by definition, the sum of its column degrees is $\deg(\det(\mathbf{A}))$, and therefore its size is at most $m^2 + m\deg(\det(\mathbf{A}))$, independently of $\mathbf{s}$.

As a simple example, consider $\mathbf{A} = \left[\begin{smallmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{B}\end{smallmatrix}\right]$ for any $\mathbf{0}$-reduced $\mathbf{B} \in \mathbb{K}[X]^{m\times m}$. Then, taking $\mathbf{s} = (0,\ldots,0,d,\ldots,d)$ with $d > 0$, $\left[\begin{smallmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{C} & \mathbf{B}\end{smallmatrix}\right]$ is an $\mathbf{s}$-reduced form of $\mathbf{A}$ for any

$\mathbf{C} \in \mathbb{K}[X]^{m\times m}$ with $\deg(\mathbf{C}) \leqslant d$; for some $\mathbf{C}$ it has size $\Theta(m^2 d)$, with d arbitrary large independently of $\deg(\mathbf{A})$.

Another important obstacle, this one independent of the shift, is that the size of the unimodular transformation leading from $\mathbf{A}$ to $\mathbf{P}$ may be beyond the target cost bound. This is the main reason why fast algorithms for $\mathbf{0}$-reduction and Hermite form computation do not directly perform unimodular transformations on $\mathbf{A}$ to reduce the degrees of its entries. Instead, they proceed in two steps: first, they work on $\mathbf{A}$ to find some equations which describe its row space, and then from these equations they reconstruct a basis of solutions to these equations in $\mathbf{0}$-reduced form or Hermite form. One may note the similarity with the two-step strategy that we outlined in Section 2.2.3 concerning the change of monomial order for a zero-dimensional ideal.

Here, we will follow a similar path to solve Problem 15 for an arbitrary shift. Yet, it seems that some new ingredient is needed, since for both Popov form and Hermite form computations, the fastest known algorithms use shift-specific properties at some point of the process. Namely, they exploit the facts that a $\mathbf{0}$-reduced form of $\mathbf{A}$ has degree at most $\deg(\mathbf{A})$ and, on the other hand, that the Hermite form of $\mathbf{A}$ is triangular. In the general case, the $\mathbf{s}$-Popov form of $\mathbf{A}$ is not necessarily triangular, and it can have degree as large as $m \deg(\mathbf{A})$.

As the first step, to find the equations we follow the idea of [GS11]: we first compute the Smith form $\mathbf{S}$ of $\mathbf{A}$ and a corresponding right unimodular transformation $\mathbf{F}$ with its columns reduced modulo the entries of $\mathbf{S}$; this is where the probabilistic aspect of the algorithm comes from. These matrices $\mathbf{S}$ and $\mathbf{F}$ give a description of the row space of $\mathbf{A}$ as the set of row vectors $\mathbf{p} \in \mathbb{K}[X]^{1\times m}$ such that $\mathbf{pF} = \mathbf{qS}$ for some $\mathbf{q} \in \mathbb{K}[X]^{1\times m}$. In other words, the row space of $\mathbf{A}$ is $\{\mathbf{p} \in \mathbb{K}[X]^{1\times m} \mid \mathbf{pF} = 0 \bmod \mathbf{S}\}$; since $\mathbf{S}$ is diagonal, this can be seen as a system of linear modular equations as in Problem 9. Then, as the second step, we efficiently find the $\mathbf{s}$-Popov solution basis for this system by relying on our algorithm for Problem 9, which is our new ingredient.

3.2.3 Deterministic computation of Hermite forms and determinants

Now, we discuss our algorithms for the fast, deterministic computation of the determinant and Hermite form of a nonsingular polynomial matrix. These results were obtained in a joint work with George Labahn and Wei Zhou [LNZ16].

Diagonal entries and determinant. The common thread in both algorithms is a method for the fast computation of the diagonal entries of a matrix triangularization; a preliminary version of this method was described in [Zho12, ZL14a]. The product of these entries gives, at least up to a constant, the determinant. Concerning the Hermite form, the degrees of the diagonal entries correspond to the shifted minimal degree of the sought matrix, which we use to design a new reduction to the problem of $\mathbf{0}$-row reduction, as detailed in Theorem 3.10 below.

In the case of determinant computation, there has been a number of efforts directed to obtaining algorithms whose complexities are given in terms of the exponent ω of matrix

multiplication. Interestingly enough, in the case of matrices over a field, it was showed in [BH74] that if there exists an algorithm for matrix multiplication for some exponent ω, then there also exists an algorithm for determinant computation with the same exponent.

In the case of the determinant of a polynomial matrix $\mathbf{A}$ with $\deg(\mathbf{A}) = d$, a recursive deterministic algorithm was given in [Sto00] making use of fraction-free Gaussian elimination with a cost of $\tilde{\mathcal{O}}(m^{\omega+1}d)$ operations. A deterministic $\mathcal{O}(m^3d^2)$ algorithm was later given in [MS03b], modifying an algorithm therein for weak Popov form computation. Using low rank perturbations, a probabilistic determinant algorithm using $\tilde{\mathcal{O}}(m^{2+\omega/2}d)$ field operations was proposed in [EGV00]. Later, [Sto03] used high order lifting to give a probabilistic algorithm which computes the determinant using $\tilde{\mathcal{O}}(m^{\omega}d)$ field operations. The algorithm in [GJV03] has a similar cost but only works on a class of generic input matrices, matrices that are well behaved in the computation.

Similarly, there has been considerable progress in the efficient computation of the Hermite form of a polynomial matrix. Algorithms with a complexity bound of $\tilde{\mathcal{O}}(m^4d)$ operations from $\mathbb{K}$ were given in [HM91] and [Ili89], where $d = \deg(\mathbf{A})$. In these references, the size of the matrices encountered during the computation is controlled by working modulo the determinant. Using matrix multiplication, the algorithms in [HM91, SL96, Vil96] reduce the cost to $\tilde{\mathcal{O}}(m^{\omega+1}d)$ operations where ω is the exponent of matrix multiplication. The algorithm in [SL96] worked with integer matrices but the results directly carry over to polynomial matrices. In [MS03b], an iterative algorithm having complexity $\mathcal{O}(m^3d^2)$ was given, thus reducing the exponent of m but at the cost of increasing that of d.

Our approach relies on an efficient method for determining the diagonal elements of a triangularization of the input matrix $\mathbf{A}$. We do this by determining a partition

$$\mathbf{U}\mathbf{A} = \begin{bmatrix} \mathbf{U}_u \\ \mathbf{U}_d \end{bmatrix} \begin{bmatrix} \mathbf{A}_\ell & \mathbf{A}_r \end{bmatrix} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ * & \mathbf{B}_2 \end{bmatrix} = \mathbf{B},$$

where $\mathbf{U}$ is a unimodular matrix, $\mathbf{A}_\ell$ has $m/2$ columns, $\mathbf{U}_u$ has $m/2$ rows, and $\mathbf{B}_1$ has dimensions $(m/2) \times (m/2)$. (The subscripts for $\mathbf{A}$ and $\mathbf{U}$ are meant to denote up, down, left, and right.) Similar transformations were used for example in algorithms for inversion [JV05, ZLS15]. Since $\mathbf{A}$ is nonsingular, $\mathbf{A}_r$ has full rank and hence $\mathbf{U}_u$ is a kernel basis for $\mathbf{A}_r$; furthermore, $\mathbf{B}_1$ is nonsingular and is therefore a basis of the row space of $\mathbf{A}_\ell$.

Then, computing such partitions for the two matrices $\mathbf{B}_1$ and $\mathbf{B}_2$ having half the dimension of $\mathbf{A}$, and continuing the process recursively until reaching dimension $m = 1$, we obtain the diagonal entries of a triangular form of $\mathbf{A}$. Up to making them monic, these are the diagonal entries of the sought Hermite form of $\mathbf{A}$.

Unfortunately, as described above, such a recursion is not necessarily efficient for our applications. In the case of determinants, $\mathbf{U}\mathbf{A}$ being lower triangular implies that we need both the product of its diagonal entries and also the determinant of the unimodular multiplier $\mathbf{U}$. For the case of Hermite form computation, a sensible approach would be to first determine a complete triangular form of $\mathbf{A}$ and then reduce the lower triangular elements using the diagonal entries with unimodular operations. In both applications it appears that we would need to know $\mathbf{U}$. However the degrees in such a unimodular multiplier can be too large for efficient computation. Indeed there are examples where the sum of the degrees in $\mathbf{U}$ is $\Theta(m^3d)$ (see Section 16.2), in which case computing $\mathbf{U}$ is beyond our target cost $\tilde{\mathcal{O}}(m^{\omega}d)$.

In order to achieve the desired efficiency, our triangularization computations need to be done without actually determining the entire unimodular matrix $\mathbf{U}$. We accomplish this by making use of shifted minimal kernel bases and column bases of polynomial matrices, whose computations can be done efficiently using algorithms from [ZLS12] and [ZL13]. Here, shifts basically help us to control the computations using row degrees rather than the degree of the polynomial matrix. Using the degree becomes an issue for efficiency when the degrees in the input matrix vary considerably from row to row. We remark that shifted minimal kernel bases and column bases, used in the context of fast block elimination, have also been used for deterministic algorithms for inversion [ZLS15] and unimodular completion [ZL14b] of polynomial matrices.

Fast algorithms for computing shifted minimal kernel bases [ZLS12] and column bases [ZL13] imply that we can deterministically find the diagonals in $\tilde{\mathcal{O}}(m^\omega \lceil s \rceil)$ field operations, where s is the average row degree of $\mathbf{A}$. We recall that the ceiling function indicates that for matrices with very low average row degree $s \in o(1)$, this cost is still $\tilde{\mathcal{O}}(m^\omega)$. By modifying this algorithm slightly we can also compute the determinant of the unimodular multiplier, giving our first contribution. Here $D_{\mathbf{A}} \leqslant s$ is the generic determinant bound as above (see Section 15.1 for more details).

Theorem 3.8. *Let $\mathbf{A}$ be a nonsingular matrix in $\mathbb{K}[X]^{m\times m}$. There is a deterministic algorithm which computes the determinant of $\mathbf{A}$ using*

$$\tilde{\mathcal{O}}(m^\omega \lceil D_{\mathbf{A}}/m \rceil) \subseteq \tilde{\mathcal{O}}(m^\omega \deg(\mathbf{A}))$$

operations in $\mathbb{K}$.

Hermite form. Applying our fast diagonal entry algorithm to the computation of the Hermite form involves more technical challenges. It solves the difficulty related to the unpredictability of the degrees in the Hermite form $\mathbf{H}$ of $\mathbf{A}$. Indeed, we know that the sum of the diagonal degrees in $\mathbf{H}$ is $\deg(\det(\mathbf{A})) \leqslant md$, and since these bound the column degrees, the sum of the degrees in $\mathbf{H}$ is $\mathcal{O}(m^2 d)$. On the other hand, the best known a priori bound for the degree of the i-th diagonal entry is $(m - i + 1)d$ and hence the sum of these bounds is $\mathcal{O}(m^2 d)$, a factor of m larger than the actual sum. Still, having the diagonal degrees, it remains a major task: that of computing the remaining entries of $\mathbf{H}$.

The probabilistic algorithm of Gupta and Storjohann [GS11, Gup11] solves the Hermite form problem using two steps, which both make use of the Smith normal form $\mathbf{S}$ of $\mathbf{A}$ and partial information on a left multiplier $\mathbf{V}$ for this Smith form. The matrices $\mathbf{S}$ and $\mathbf{V}$ can be computed with a Las Vegas randomized algorithm using an expected number of $\tilde{\mathcal{O}}(m^\omega d)$ field operations [GS11, Gup11], relying in particular on high-order lifting [Sto03, Section 17]. The first step of their algorithm consists of computing the diagonal entries of $\mathbf{H}$ by triangularization of a $2m \times 2m$ matrix involving $\mathbf{S}$ and $\mathbf{V}$, a computation done in $\tilde{\mathcal{O}}(m^\omega d)$ operations [Gup11]. The second step sets up a system of linear modular equations which admits $\mathbf{A}$ as a basis of solutions: the matrix of the system is $\mathbf{V}$ and the moduli are the diagonal entries of $\mathbf{S}$. The degrees of the diagonal entries obtained in the first step are then used to find $\mathbf{H}$ as another basis of solutions of this system, computed in $\tilde{\mathcal{O}}(m^\omega d)$ [GS11] using in particular fast minimal approximant basis and partial linearization techniques [Sto06, ZL12].

The algorithm presented here for Hermite forms follows a two-step process similar to the algorithm of Gupta and Storjohann, but it avoids using the Smith form of $\mathbf{A}$, whose deterministic computation in $\tilde{\mathcal{O}}(m^\omega d)$ still remains an open problem. Instead, as explained above, we compute the diagonal entries of $\mathbf{H}$ deterministically using $\tilde{\mathcal{O}}(m^\omega \lceil s \rceil)$ field operations, where s is the average of the column degrees of $\mathbf{A}$. As for the second step, we note that the tuple of diagonal degrees of $\mathbf{H}$ also coincide with the $\mathbf{h}$-minimal degree of the row space of $\mathbf{A}$, where $\mathbf{h}$ is the Hermite shift described above. Knowing this minimal degree, we use the ideas in Section 1.2.1 as well as partial linearization techniques from [GSSV12, Section 6] to show that $\mathbf{H}$ can then be computed via a single call to fast deterministic row reduction [GSSV12] using $\tilde{\mathcal{O}}(m^\omega d)$ field operations. This new problem reduction illustrates the folklore fact that knowing in advance the degree shape of reduced or normal forms makes their computation much easier, something already observed and exploited in [GS11, Zho12] and Section 1.2.1 and Chapter 2.

This approach results in a deterministic $\tilde{\mathcal{O}}(m^\omega d)$ algorithm for Hermite form computation, which is satisfactory for matrices $\mathbf{A}$ that have most entries of similar degree $d = \deg(\mathbf{A})$. Using the input degree reduction summarized in Theorem 3.5, we obtain the following result.

Theorem 3.9. *Let $\mathbf{A}$ be a nonsingular matrix in $\mathbb{K}[X]^{m\times m}$. There is a deterministic algorithm which computes the Hermite form of $\mathbf{A}$ using*

$$\tilde{\mathcal{O}}(m^\omega \lceil D_{\mathbf{A}}/m \rceil) \subseteq \tilde{\mathcal{O}}(m^\omega \deg(\mathbf{A}))$$

operations in $\mathbb{K}$.

Concerning the computation of the Hermite form with known diagonal degrees, our results can be straightforwardly generalized to the case of shifted Popov form computation, with only a few minor modifications. Formally, this leads to the following result (the hidden logarithmic factors in the cost bound are exactly those in [GSSV12, Theorem 18]).

Theorem 3.10. *Let $\mathbf{A}$ be a nonsingular matrix in $\mathbb{K}[X]^{m\times m}$, and let $\mathbf{s} \in \mathbb{Z}^m$. Let $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$ denote the $\mathbf{s}$-minimal degree of the row space of $\mathbf{A}$, which is also the tuple of diagonal degrees of the $\mathbf{s}$-Popov form of $\mathbf{A}$. If $\boldsymbol{\delta}$ is known, then one can compute the $\mathbf{s}$-Popov form of $\mathbf{A}$ deterministically using $\tilde{\mathcal{O}}(m^\omega \deg(\mathbf{A}))$ operations in $\mathbb{K}$.*

We note however that, as of today and to the best of our knowledge, the Hermite form is the only case of shifted Popov forms for which there is a known algorithm to efficiently compute the degrees of the diagonal entries separately from the computation of the whole form.

Part II

Relation bases for arbitrary multiplication matrices

Contents

Chapter 4 Computing relation bases via linear algebra **119**

4.1 The linear algebra viewpoint 120

4.1.1 Linearization: viewing polynomial relations as scalar relations . 120

4.1.2 Bounded-degree relations and nullspace of multi-Krylov matrices 122

4.1.3 Multi-Krylov matrices in the univariate case 125

4.2 Fast computation of the monomial basis 126

4.2.1 Row rank profile and monomial basis 127

4.2.2 Structure and row rank profile of a multi-Krylov matrix 128

4.2.3 Computing the row rank profile of a multi-Krylov matrix 129

4.3 Fast computation of the relation basis 134

4.3.1 Simultaneous computation of normal forms of monomials 134

4.3.2 Univariate case: computing shifted Popov relation bases 136

4.3.3 Computing reduced Gröbner relation bases 138

Chapter 5 Computing multiplication matrices from a Gröbner basis **141**

5.1 Structural properties of the monomial basis 141

5.2 The case of two variables 142

5.3 Computing rows of a Krylov matrix 144

5.4 Computing the multiplication matrices 144

4

Computing relation bases via linear algebra

In this chapter, we use techniques from linear algebra over $\mathbb{K}$ to solve Problems 2 and 4 efficiently. The basic tool is a linearization of the problem, referring to an interpretation of operations on polynomials as operations from $\mathbb{K}$-linear algebra. In this framework, a polynomial is seen as a coefficient vector, and multiplication by a variable corresponds to multiplying this coefficient vector by the multiplication matrix of this variable. Concerning the computation of generating sets of relation modules as in Problem 2, the input is already in linearized form, yet the output is not. Here, we will first complete the linearization in Section 4.1, leading to a correspondence between relations of bounded degree and vectors in the nullspace of some structured matrix which is called a *multi-Krylov matrix*.

Linearization is a ubiquitous tool in computations with polynomials. For example, a well-known particular case of the above property is that, for two polynomials $f, g \in \mathbb{K}[X]$, the solutions $p, q \in \mathbb{K}[X]$ to the Bézout equation $pf + qg = 0$ with $\deg(p) < \deg(g)$ and $\deg(q) < \deg(f)$ precisely correspond to the nullspace of the Sylvester matrix [Syl53] of f and g. Concerning multivariate polynomials, many algorithms adopt a linear algebra point of view to rely on computations with Macaulay matrices [Mac02, Mac16]. Krylov matrices were used to compute Popov and Hermite bases of relations in [Kai80, Chapter 6]; besides, one can interpret the change of monomial order algorithm of [FGLM93] as the search of a specific collection of vectors in the nullspace of a multi-Krylov matrix.

To identify this collection, we will order the rows of the latter matrix according to the monomial order in input of Problem 2. Then, the polynomials in the sought Gröbner relation basis involve in priority the first rows of the matrix. More precisely, in Section 4.2 we show that the row rank profile of the multi-Krylov matrix correspond to the monomial basis of the quotient module. To compute this monomial basis efficiently, we extend ideas from [KG85] to this more general context. This can be seen as a way of both introducing matrix multiplication in the algorithm and taking into account some structure of the matrix by constantly considering only a small subset of its rows.

Finally, we exploit the knowledge of the monomial basis to compute the reduced Gröbner relation basis. Our algorithm is efficient for arbitrary multiplication matrices and an arbitrary monomial order. In univariate contexts, this translates as a fast algorithm for computing the shifted Popov relation basis for any multiplication matrix and shift.

4.1 The linear algebra viewpoint

In this section, we explain how polynomial relations as in Definition 2.4 can be interpreted in terms of linear algebra. We rely on a *linearization* of the relations: having fixed some degree bounds, these polynomials are seen as finite lists of coefficients, or in other words, row vectors over $\mathbb{K}$. Then, we will show how, from matrices $\mathbf{M}$ and $\mathbf{F}$ as in the input of Problem 2, one can build a matrix over $\mathbb{K}$ whose left nullspace contains all relations of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ up to the chosen degree bounds.

4.1.1 Linearization: viewing polynomial relations as scalar relations

As an introduction to linearization techniques, we first describe a specific linearization of relations in the univariate case, as in Problem 4. Let $\mathbf{M} \in \mathbb{K}^{D\times D}$ and $\mathbf{F} \in \mathbb{K}^{m\times D}$, and fix some degree bound $\beta \in \mathbb{Z}_{>0}$. In what follows, we let $\mathbb{K}[X]_{<\beta}$ denote the set of polynomials in $\mathbb{K}[X]$ of degree less than β, and we focus on relations of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ which have degree less than β; thus, these are row vectors in $\mathbb{K}[X]^{1\times m}_{<\beta}$.

First, the operation of *expansion* allows us to see a polynomial vector of bounded degree as a scalar vector formed by its coefficients in the field. Given $\mathbf{p} \in \mathbb{K}[X]^{1\times m}_{<\beta}$, we write it as a polynomial of vectors: $\mathbf{p} = \mathbf{p}_0 + \mathbf{p}_1 X + \cdots + \mathbf{p}_{\beta-1}X^{\beta-1}$ where each $\mathbf{p}_j$ is a scalar vector in $\mathbb{K}^{1\times m}$. Then, the expansion of $\mathbf{p}$ in degree β is the vector formed by concatenation as $\mathcal{E}_\beta(\mathbf{p}) = [\mathbf{p}_0 \mid \mathbf{p}_1 \mid \cdots \mid \mathbf{p}_{\beta-1}] \in \mathbb{K}^{1\times m\beta}$.

The reciprocal operation is called *compression*, and allows us to transform back a scalar vector into a polynomial one. Given a vector $\mathbf{v} \in \mathbb{K}^{1\times m\beta}$, we write it with blocks $\mathbf{v} = [\mathbf{v}_0 \mid \mathbf{v}_1 \mid \cdots \mid \mathbf{v}_{\beta-1}]$ where each $\mathbf{v}_j$ is in $\mathbb{K}^{1\times m}$, and then we define its compression in degree β as $\mathcal{C}_\beta(\mathbf{v}) = \mathbf{v}_0 + \mathbf{v}_1 X + \cdots + \mathbf{v}_{\beta-1}X^{\beta-1} \in \mathbb{K}[X]^{1\times m}_{<\beta}$.

Now, given some matrices $\mathbf{M} \in \mathbb{K}^{D\times D}$ and $\mathbf{F} \in \mathbb{K}^{m\times D}$, our problem asks to find vectors $\mathbf{p} \in \mathbb{K}[X]^{1\times m}$ such that, in particular, $\mathbf{p}\cdot\mathbf{F} = 0$. Assuming that $\deg(\mathbf{p}) < \beta$, and writing $\mathbf{p} = \mathbf{p}_0 + \mathbf{p}_1 X + \cdots + \mathbf{p}_{\beta-1}X^{\beta-1}$, we recall that $\mathbf{p}\cdot\mathbf{F} = \mathbf{p}_0\mathbf{F} + \mathbf{p}_1\mathbf{F}\mathbf{M} + \cdots + \mathbf{p}_{\beta-1}\mathbf{F}\mathbf{M}^{\beta-1}$. Then, in accordance to the expansion of $\mathbf{p}$, the input $(\mathbf{M}, \mathbf{F})$ is expanded as follows:

$$\mathcal{K}_\beta(\mathbf{M}, \mathbf{F}) = \begin{bmatrix} \mathbf{F} \\ \mathbf{F}\mathbf{M} \\ \vdots \\ \mathbf{F}\mathbf{M}^{\beta-1} \end{bmatrix} \in \mathbb{K}^{m\beta\times D} \tag{4.1}$$

so as to ensure that $\mathbf{p}\cdot\mathbf{F} = \mathcal{E}_\beta(\mathbf{p})\mathcal{K}_\beta(\mathbf{M}, \mathbf{F})$ for any $\mathbf{p} \in \mathbb{K}[X]^{1\times m}_{<\beta}$. In particular, $\mathbf{p}$ is a relation of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ if and only if $\mathcal{E}_\beta(\mathbf{p})\mathcal{K}_\beta(\mathbf{M}, \mathbf{F}) = 0$, that is, its expansion $\mathcal{E}_\beta(\mathbf{p})$ is in the left nullspace of $\mathcal{K}_\beta(\mathbf{M}, \mathbf{F})$.

Example 4.1. In this example, we have $m = D = 3$ and the base field is the finite field with 97 elements; the input matrices are

$$\mathbf{F} = \begin{bmatrix} 27 & 49 & 29 \\ 50 & 58 & 0 \\ 77 & 10 & 29 \end{bmatrix} \qquad \text{and} \qquad \mathbf{Z} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}.$$

Since the minimal polynomial of $\mathbf{Z}$ is X^3, we choose $\beta = 4$, thus ensuring that all relations in the Popov basis will have degree less than β. Then, we have

$$\mathcal{K}_\beta(\mathbf{Z}, \mathbf{F}) = \begin{bmatrix} 27 & 49 & 29 \\ 50 & 58 & 0 \\ 77 & 10 & 29 \\ 0 & 27 & 49 \\ 0 & 50 & 58 \\ 0 & 77 & 10 \\ 0 & 0 & 27 \\ 0 & 0 & 50 \\ 0 & 0 & 77 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

It is easily checked that $\mathbf{p}_1 = [-1 \quad -1 \quad 1] \in \mathbb{K}[X]^{1\times m}$ is a relation of $\mathrm{Syz}_{\mathbf{Z}}(\mathbf{F})$, since $\mathbf{F}_{3,*} = \mathbf{F}_{1,*} + \mathbf{F}_{2,*}$. Other relations are for example $\mathbf{p}_2 = [3X + 13 \quad X + 57 \quad 0]$ which has row degree 1, $\mathbf{p}_3 = [X^2 + 40X + 82 \quad 76 \quad 0]$ which has row degree 2, and $\mathbf{p}_4 = [X^3 \quad 0 \quad 0]$ which has row degree 3. We have

$$\begin{array}{rcl} \mathcal{E}(\mathbf{p}_1) & = & [\ 96 \ \ 96 \ \ 1 \ \mid \ 0 \ \ 0 \ \ 0 \ \mid \ 0 \ \ 0 \ \ 0 \ \mid \ 0 \ \ 0 \ \ 0\ ] \\ \mathcal{E}(\mathbf{p}_2) & = & [\ 13 \ \ 57 \ \ 0 \ \mid \ 3 \ \ 1 \ \ 0 \ \mid \ 0 \ \ 0 \ \ 0 \ \mid \ 0 \ \ 0 \ \ 0\ ] \\ \mathcal{E}(\mathbf{p}_3) & = & [\ 82 \ \ 76 \ \ 0 \ \mid \ 40 \ \ 0 \ \ 0 \ \mid \ 1 \ \ 0 \ \ 0 \ \mid \ 0 \ \ 0 \ \ 0\ ] \\ \mathcal{E}(\mathbf{p}_4) & = & [\ 0 \ \ 0 \ \ 0 \ \mid \ 0 \ \ 0 \ \ 0 \ \mid \ 0 \ \ 0 \ \ 0 \ \mid \ 1 \ \ 0 \ \ 0\ ]. \end{array}$$

Besides, one can verify, for example using a computer algebra system, that the matrix

$$\mathbf{P} = \begin{bmatrix} \mathbf{p}_3 \\ \mathbf{p}_2 \\ \mathbf{p}_1 \end{bmatrix} = \begin{bmatrix} X^2 + 40X + 82 & 76 & 0 \\ 3X + 13 & X + 57 & 0 \\ 96 & 96 & 1 \end{bmatrix},$$

is the Popov relation basis of $\mathrm{Syz}_{\mathbf{Z}}(\mathbf{F})$; note that the latter is the module of Hermite-Padé approximants of order 3 for

$$\mathbf{F} \begin{bmatrix} 1 \\ X \\ X^2 \end{bmatrix} = \begin{bmatrix} 29X^2 + 49X + 27 \\ 58X + 50 \\ 29X^2 + 10X + 77 \end{bmatrix}. \qquad ✍$$

Now, having built a matrix over $\mathbb{K}$ whose left nullspace corresponds to relations, we would like tools to interpret in this linearized framework the shifted degree minimality of the sought relation bases. Observing Example 4.1, we see that $\mathbf{p}_1$ has $\mathbf{0}$-degree 0, which makes it a perfect candidate to appear in a $\mathbf{0}$-minimal relation basis such as $\mathbf{P}$; conversely, $\mathbf{p}_4$ has $\mathbf{0}$-degree 3, and no row has degree more than 2 in $\mathbf{P}$. On the other hand, if we consider the shift $\mathbf{s} = (0, 3, 6)$, then the $\mathbf{s}$-degree of $\mathbf{p}_4$ is 3 while the one of $\mathbf{p}_1$ is 6. Therefore, if one is looking for the rows of an $\mathbf{s}$-minimal relation basis, $\mathbf{p}_4$ is a better candidate than $\mathbf{p}_1$.

We see through this example that the uniform shift $\mathbf{0}$ leads to look for elements of the nullspace of $\mathcal{K}_\beta(\mathbf{Z}, \mathbf{F})$ which involve in priority the first rows of the matrix, while the shift $\mathbf{s} = (0, 3, 6)$ leads to look for elements of the nullspace which involve in priority the rows $\mathbf{F}_{1,*}$, $\mathbf{F}_{1,*}\mathbf{Z}$, $\mathbf{F}_{1,*}\mathbf{Z}^2$, and $\mathbf{F}_{1,*}\mathbf{Z}^3$, before considering any of the rows $\mathbf{F}_{2,*}\mathbf{Z}^j$ and $\mathbf{F}_{3,*}\mathbf{Z}^j$ for $0 \leqslant j < \beta$.

Thus, to take the shift into account, we will define below a permutation of the rows of $\mathcal{K}_\beta(\mathbf{M}, \mathbf{F})$ such that a basis of its nullspace which involve in priority its first rows correspond to a shifted minimal relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$. This permutation is naturally linked to the monomial order on $\mathbb{K}[X]^{1\times m}$ induced by the shift. In the next section, we present the linearization framework with more generality and in the multivariate context, proving some assertions that have been made here without details. Then, we will come back to the univariate case in Section 4.1.3 to give the details of the construction of $\mathcal{K}_\beta(\mathbf{M}, \mathbf{F})$ when one is given a shift rather than a monomial order.

4.1.2 Bounded-degree relations and nullspace of multi-Krylov matrices

Here, we generalize the construction of the striped Krylov matrix above to several variables and to an arbitrary monomial order. This results in a type of matrix with several layers of structure that we call *multi-Krylov*. Then, we give a precise link between left nullspaces of such matrices and relations of bounded degree.

Let $r \in \mathbb{Z}_{>0}$ and consider the polynomial ring $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X_1, \ldots, X_r]$. We fix an interger $m \in \mathbb{Z}_{>0}$, and we see elements of $\mathbb{K}[\boldsymbol{X}]^m$ as row vectors in $\mathbb{K}[\boldsymbol{X}]^{1\times m}$.

We first describe the expansion and compression operation, to convert polynomials of bounded degrees into their coefficient vectors and vice versa. These operations naturally depend on the choice of a monomial order, to choose in which order one should concatenate the coefficients of a polynomial so as to obtain its coefficient vector.

For example, we made such a choice implicitly in Section 4.1.1 when defining the expansion $\mathcal{E}_\beta(\mathbf{p})$ of a univariate row vector $\mathbf{p}$: we concatenated the constant terms of $\mathbf{p}$ first, then its terms of degree 1, etc. Thus, the terms of $\mathbf{p}$ were implicitly arranged in $\mathcal{E}_\beta(\mathbf{p})$ according to the term-over-position order.

Then, let $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^{1\times m}$, and let $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_r) \in \mathbb{Z}^r_{>0}$ be some degree bounds. To define and use the linearization operations, it will be convenient to rely on the following indexing function.

Definition 4.2 ($(\prec, \boldsymbol{\beta})$-indexing)**.** *Let $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_r) \in \mathbb{Z}^m_{>0}$ and let $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^{1\times m}$. Then, we define the $(\prec, \boldsymbol{\beta})$-indexing function as the unique bijection*

$$\phi_{\prec,\boldsymbol{\beta}} : \{\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i, \mathbf{0} \leqslant \mathbf{e} < \boldsymbol{\beta}, 1 \leqslant i \leqslant m\} \to \{1, \ldots, m\beta_1 \cdots \beta_r\}$$

which is increasing for $\prec$, that is, $\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i \prec \boldsymbol{X}^{\mathbf{e}'}\mathbf{c}_{i'}$ if and only if $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i) < \phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}'}\mathbf{c}_{i'})$.

We now define the expansion and compression operations, and then we give an example with the lexicographic term-over-position order in two variables.

In what follows, we denote by $\mathbb{K}[\boldsymbol{X}]_{<\boldsymbol{\beta}}$ the set of polynomials $p \in \mathbb{K}[\boldsymbol{X}]$ such that $\deg_{X_k}(p) < \beta_k$ for $1 \leqslant k \leqslant r$. Then, we have a correspondence between bounded-degree polynomials and row vectors, as follows:

$$\begin{array}{ccc} \text{polynomial } \mathbf{p} \in \mathbb{K}[\boldsymbol{X}]^{1\times m}_{<\boldsymbol{\beta}} & \xrightarrow{\text{expansion}} & \text{vector } \mathbf{v} = [v_k]_k \in \mathbb{K}^{1\times m\beta_1\cdots\beta_r} \text{ with} \\ \mathbf{p} = \sum_j u_j \boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j} \text{ with } u_j \neq 0 & \xleftarrow[\text{compression}]{} & v_k = u_j \text{ for } k = \phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j}). \end{array}$$

That is, for a polynomial $\mathbf{p} \in \mathbb{K}[\boldsymbol{X}]^{1\times m}_{<\boldsymbol{\beta}}$, the *expansion* of $\mathbf{p}$ in degree $\boldsymbol{\beta}$ and with respect to $\prec$ is the vector $\mathcal{E}_{\prec,\boldsymbol{\beta}}(\mathbf{p}) \in \mathbb{K}^{1\times m\beta_1\cdots\beta_r}$ whose entry at index $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i)$ is the coefficient of the term involving $\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i$ in $\mathbf{p}$. Conversely, given a vector $\mathbf{v} \in \mathbb{K}^{1\times m\beta_1\cdots\beta_r}$, we define its *compression* in degree $\boldsymbol{\beta}$ and with respect to $\prec$ as being the polynomial $\mathcal{C}_{\prec,\boldsymbol{\beta}}(\mathbf{v}) \in \mathbb{K}[\boldsymbol{X}]^{1\times m}_{<\boldsymbol{\beta}}$ such that the coefficient of its term involving the monomial $\phi^{-1}_{\prec,\boldsymbol{\beta}}(k)$ is the entry at index k of $\mathbf{v}$.

Example 4.3. Let us consider bivariate polynomials in $\mathbb{K}[X,Y]$, where $\mathbb{K}$ is the finite field with 97 elements. Let $\prec_{\text{lex}}$ be the lexicographic order on $\mathbb{K}[X,Y]$ with $Y \prec_{\text{lex}} X$, and let $\prec = \prec^{\text{top}}_{\text{lex}}$ be the $\prec_{\text{lex}}$-term over position order on $\mathbb{K}[X,Y]^{1\times 2}$. We choose the degree bounds $\boldsymbol{\beta} = (2,3)$.

Following Definition 4.2, we order the monomials

$$\{X^jY^k\mathbf{c}_i, \mathbf{0} \leqslant (j,k) < (2,3), 1 \leqslant i \leqslant 2\}$$

increasingly according to $\prec^{\text{top}}_{\text{lex}}$, which yields

Monomial $X^jY^k\mathbf{c}_i$	Index $\phi_{\prec^{\text{top}}_{\text{lex}},(2,3)}(X^jY^k\mathbf{c}_i)$
$\begin{bmatrix}1 & 0\end{bmatrix}$	1
$\begin{bmatrix}0 & 1\end{bmatrix}$	2
$\begin{bmatrix}Y & 0\end{bmatrix}$	3
$\begin{bmatrix}0 & Y\end{bmatrix}$	4
$\begin{bmatrix}Y^2 & 0\end{bmatrix}$	5
$\begin{bmatrix}0 & Y^2\end{bmatrix}$	6
$\begin{bmatrix}X & 0\end{bmatrix}$	7
$\begin{bmatrix}0 & X\end{bmatrix}$	8
$\begin{bmatrix}XY & 0\end{bmatrix}$	9
$\begin{bmatrix}0 & XY\end{bmatrix}$	10
$\begin{bmatrix}XY^2 & 0\end{bmatrix}$	11
$\begin{bmatrix}0 & XY^2\end{bmatrix}$	12

Let $\mathbf{p}$ be the polynomial in $\mathbb{K}[X,Y]^{1\times 2}_{<(2,3)}$ and $\mathbf{v}$ be the vector in $\mathbb{K}^{1\times 12}$ defined by

$$\begin{aligned} \mathbf{p} &= [46 + 95Y + 75X + 10XY\ ,\ 36 + 18Y + 38Y^2 + 77X + 83XY + 35XY^2] \\ \mathbf{v} &= \begin{bmatrix} 86 & 0 & 32 & 83 & 54 & 26 & 0 & 68 & 86 & 0 & 54 & 22 \end{bmatrix} \end{aligned}$$

In this case, the expansion of $\mathbf{p}$ and the compression of $\mathbf{v}$ are given by

$$\mathcal{E}_{\prec,\boldsymbol{\beta}}(\mathbf{p}) = \begin{bmatrix} 46 & 36 & 95 & 18 & 0 & 38 & 75 & 77 & 10 & 83 & 0 & 35 \end{bmatrix}$$
$$\mathcal{C}_{\prec,\boldsymbol{\beta}}(\mathbf{v}) = [86 + 32Y + 54Y^2 + 86XY + 54XY^2\ ,\ 83Y + 26Y^2 + 68X + 22XY^2]\ . \quad ✍$$

Now, we detail the construction of the multi-Krylov matrix. Let $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$ be pairwise commuting matrices in $\mathbb{K}^{D\times D}$, and let $\mathbf{F} \in \mathbb{K}^{m\times D}$. From the module structure induced by the multiplication matrices as described in Sections 1.3.5 and 2.1, for a polynomial $\mathbf{p} = [p_1, \ldots, p_m] \in \mathbb{K}[\boldsymbol{X}]^{1\times m}$ we have

$$\begin{aligned} \mathbf{p}\cdot\mathbf{F} &= p_1\cdot\mathbf{F}_{1,*} + \cdots + p_m\cdot\mathbf{F}_{m,*} \\ &= \mathbf{F}_{1,*}p_1(\mathbf{M}) + \cdots + \mathbf{F}_{m,*}p_m(\mathbf{M}), \end{aligned}$$

where we write $p_i(\mathbf{M})$ for $p_i(\mathbf{M}_1, \ldots, \mathbf{M}_r)$. As a result, $\mathbf{p}$ being a relation of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ means that the coefficients of $\mathbf{p}$ form a $\mathbb{K}$-linear combination of vectors of the form $\mathbf{F}_{i,*}\mathbf{M}^{\mathbf{e}}$ which is zero. If furthermore $\mathbf{p}$ is nonzero and has its degrees in each variable bounded by $(\beta_1, \ldots, \beta_r)$, then it corresponds to a nontrivial $\mathbb{K}$-linear relation between the row vectors

$$\{\mathbf{F}_{i,*}\mathbf{M}^{\mathbf{e}},\ \mathbf{0} \leqslant \mathbf{e} < \boldsymbol{\beta}, 1 \leqslant i \leqslant m\}.$$

This leads us to consider the following matrices, formed by these row vectors ordered according to $\phi_{\prec,\boldsymbol{\beta}}$.

Definition 4.4 (Multi-Krylov matrix)**.** *Let $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r) \in \mathbb{K}^{D\times D}$ be pairwise commuting matrices, let $\mathbf{F} \in \mathbb{K}^{m\times D}$, let $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_r) \in \mathbb{Z}^r_{>0}$, and let $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^{1\times m}$. The* $(\prec, \boldsymbol{\beta})$-multi-Krylov matrix for $(\mathbf{M}, \mathbf{F})$*, denoted by $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M}, \mathbf{F})$, is defined as the matrix in $\mathbb{K}^{m\beta_1\cdots\beta_r\times D}$ whose row at index $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i)$ is $\mathbf{F}_{i,*}\mathbf{M}^{\mathbf{e}}$.*

Example 4.5. We place ourselves in the situation of Example 4.3, with two variables X and Y, the dimension $m = 2$, the $\prec_{\text{lex}}$-TOP order on $\mathbb{K}[X, Y]^{1\times 2}$ with $Y \prec_{\text{lex}} X$, and degree bounds $\boldsymbol{\beta} = (2, 3)$.

We further consider the vector space dimension $D = 3$. Then, let $\mathbf{M} = (\mathbf{M}_X, \mathbf{M}_Y)$ be matrices in $\mathbb{K}^{3\times 3}$ such that $\mathbf{M}_X\mathbf{M}_Y = \mathbf{M}_Y\mathbf{M}_X$, and let $\mathbf{F}$ be some matrix in $\mathbb{K}^{2\times 3}$.

In this case, from the indexing function $\phi_{\prec^{\text{top}}_{\text{lex}},(2,3)}$ described in Example 4.1 it follows that the multi-Krylov matrix for $(\mathbf{M}, \mathbf{F})$ is

$$\mathcal{K}_{\prec^{\text{top}}_{\text{lex}},(2,3)}(\mathbf{M}, \mathbf{F}) = \begin{bmatrix} \mathbf{F} \\ \mathbf{F}\mathbf{M}_Y \\ \mathbf{F}\mathbf{M}_Y^2 \\ \mathbf{F}\mathbf{M}_X \\ \mathbf{F}\mathbf{M}_Y\mathbf{M}_X \\ \mathbf{F}\mathbf{M}_Y^2\mathbf{M}_X \end{bmatrix} \in \mathbb{K}^{12\times 3}. \quad ✍$$

By construction, we have the following result which relates the left nullspace of the multi-Krylov matrix with the set of bounded-degree relations.

Lemma 4.6. *If $\mathbf{v} \in \mathbb{K}^{1\times m\beta_1\cdots\beta_r}$ is in the left nullspace of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M}, \mathbf{F})$, then $\mathcal{C}_{\prec,\boldsymbol{\beta}}(\mathbf{v}) \in \mathbb{K}[\boldsymbol{X}]^{1\times m}_{<\boldsymbol{\beta}}$ is a relation of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$. Conversely, if $\mathbf{p} \in \mathbb{K}[\boldsymbol{X}]^{1\times m}_{<\boldsymbol{\beta}}$ is a relation of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$, then $\mathcal{E}_{\prec,\boldsymbol{\beta}}(\mathbf{p}) \in \mathbb{K}^{1\times m\beta_1\cdots\beta_r}$ is in the left nullspace of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M}, \mathbf{F})$.*

4.1.3 Multi-Krylov matrices in the univariate case

Going back to the univariate case, we show how to construct the indexing function and thus the multi-Krylov matrix when the monomial order is given by a shift. Here, we have one multiplication matrix $\mathbf{M} \in \mathbb{K}^{D\times D}$, a matrix $\mathbf{F} \in \mathbb{K}^{m\times D}$, a shift $\mathbf{s} \in \mathbb{Z}^m$, and a degree bound $\beta \in \mathbb{Z}_{>0}$.

We have seen in Example 1.38 and Section 1.3.4 that $\mathbf{s}$ defines a monomial order $<^{\mathbf{s}\text{-top}}$ on $\mathbb{K}[X]^{1\times m}$, which we called a shifted term-over-position order. This directly allows one to use the definition of multi-Krylov matrices in the previous section. Still, for the sake of clarity, we detail here how the indexing function of Definition 4.2 can be deduced from the shift $\mathbf{s}$.

In Section 4.1.1, we defined a specific multi-Krylov matrix $\mathcal{K}_\beta(\mathbf{M},\mathbf{F})$ in Eq. (4.1). Its rows are $\mathbf{F}_{i,*}\mathbf{M}^e$ for $0 \leqslant e < \beta$ and $1 \leqslant i \leqslant m$, ordered by lexicographically increasing pairs (e,i). More generally, when an arbitrary shift is given in input, we would like the multi-Krylov matrix to be some permutation of these rows so as to reflect their priority, which is induced by the shift.

This priority is as follows. Let $\mathbf{v} \in \mathbb{K}^{1\times m\beta}$ be any linear relation between the rows of $\mathcal{K}_\beta(\mathbf{M},\mathbf{F})$ involving the row $\mathbf{F}_{i,*}\mathbf{M}^e$; this means that $\mathcal{C}_\beta(\mathbf{v}) = \mathbf{p}_0+\mathbf{p}_1X+\cdots+\mathbf{p}_{\beta-1}X^{\beta-1}$ is a relation of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ and the coefficient in column i of $\mathbf{p}_e$ is nonzero. Then, this implies that the $\mathbf{s}$-degree of $\mathbf{p}$ is at least $s_i + e$. As a consequence, since the $\mathbf{s}$-degree is precisely what we want to minimize in order to obtain an $\mathbf{s}$-minimal relation basis, the priority of the rows of $\mathcal{K}_\beta(\mathbf{M},\mathbf{F})$ can be measured by the function $(e,i) \mapsto s_i + e$.

Our goal is to define an indexing function $\phi_{\mathbf{s},\beta}(e,i)$ which we will use to specify the order of the rows $\mathbf{F}_{i,*}\mathbf{M}^e$ in the multi-Krylov matrix $\mathcal{K}_{\mathbf{s},\beta}(\mathbf{M},\mathbf{F})$. Then, the previous paragraph indicates that this function should be such that $\phi_{\mathbf{s},\beta}(e,i) < \phi_{\mathbf{s},\beta}(e',i')$ whenever $s_i + e < s_{i'} + e'$. This is not enough to define $\phi_{\mathbf{s},\beta}$: we still need to break ties for pairs $(e,i) \neq (e',i')$ which are such that $s_i+e = s_{i'}+e'$. In the latter case, we arbitrarily choose that $\phi_{\mathbf{s},\beta}(e,i) < \phi_{\mathbf{s},\beta}(e',i')$ if $i < i'$.

To summarize, in this univariate context the indexing function is the unique bijection

$$\phi_{\mathbf{s},\beta} : \{0,\dots,\beta-1\} \times \{1,\dots,m\} \to \{1,\dots,m\beta\}$$

such that

(i) if $s_i + e < s_{i'} + e'$ then $\phi_{\mathbf{s},\beta}(e,i) < \phi_{\mathbf{s},\beta}(e',i')$;

(ii) if $s_i + e = s_{i'} + e'$ and $i < i'$ then $\phi_{\mathbf{s},\beta}(e,i) < \phi_{\mathbf{s},\beta}(e',i')$.

for all $0 \leqslant e, e' < \beta$ and $1 \leqslant i, i' \leqslant m$.

Remark 4.7. This is naturally the indexing function that one would obtain using Definition 4.2 along with the remarks in Example 1.38 and Section 1.3.4 concerning the monomial order $<^{\mathbf{s}\text{-top}}$ corresponding to the shift $\mathbf{s}$. We note also that the arbitrary tie-breaking above is consistent with the shifted TOP order, and also with the choice made in the definition of $\mathbf{s}$-pivots where one considers as a leading term the *rightmost* one among those which have the highest $\mathbf{s}$-degree. ☕

Then, the linearization operations are as follows. For a polynomial $\mathbf{p} = [p_i]_i \in \mathbb{K}[X]^{1\times m}_{<\beta}$, the shifted expansion $\mathcal{E}_{\mathbf{s},\beta}(\mathbf{p})$ is the vector in $\mathbb{K}^{1\times m\beta}$ whose column $\phi_{\mathbf{s},\beta}(i,e)$ is formed by the coefficient of degree e in p_i, for all $0 \leqslant e < \beta$ and $1 \leqslant i \leqslant m$. For a scalar vector $\mathbf{v} \in \mathbb{K}^{1\times m\beta}$, the shifted compression $\mathcal{C}_{\mathbf{s},\beta}(\mathbf{v})$ is the polynomial vector in $\mathbb{K}[X]^{1\times m}_{<\beta}$ obtained by the inverse operation.

Furthermore, the multi-Krylov matrix $\mathcal{K}_{\mathbf{s},\beta}(\mathbf{M},\mathbf{F})$ is the matrix in $\mathbb{K}^{m\beta\times D}$ whose row $\phi_{\mathbf{s},\beta}(e,i)$ is $\mathbf{F}_{i,*}\mathbf{M}^e$ for all $0 \leqslant e < \beta$ and $1 \leqslant i \leqslant m$. This is a specific case of the multi-Krylov matrices of Definition 4.4, so that all properties that are given for the general multi-Krylov matrices will also hold for the ones defined from a shift as presented here. For example, Lemma 4.6 states that the left nullspace of $\mathcal{K}_{\mathbf{s},\beta}(\mathbf{M},\mathbf{F})$ corresponds to the set of relations of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ which have degree less than β.

In the specific case of the uniform shift $\mathbf{s} = \mathbf{0}$, we have $\phi_{\mathbf{0},\beta}(e,i) = i + me$, and using notation from Section 4.1.1, we have the identities $\mathcal{K}_{\mathbf{0},\beta}(\mathbf{M},\mathbf{F}) = \mathcal{K}_\beta(\mathbf{M},\mathbf{F})$, $\mathcal{C}_{\mathbf{0},\beta}(\mathbf{M}) = \mathcal{C}_\beta(\mathbf{M})$, $\mathcal{E}_{\mathbf{0},\beta}(\mathbf{P}) = \mathcal{E}_\beta(\mathbf{P})$.

Example 4.8 (Example 4.1 continued). In the context of Example 4.1, if we consider the shifts $\mathbf{s} = (0,3,6)$ and $\mathbf{t} = (3,0,2)$, we have

$$\mathcal{K}_{\mathbf{s},\beta}(\mathbf{Z},\mathbf{F}) = \begin{bmatrix} 27 & 49 & 29 \\ 0 & 27 & 49 \\ 0 & 0 & 27 \\ 0 & 0 & 0 \\ 50 & 58 & 0 \\ 0 & 50 & 58 \\ 0 & 0 & 50 \\ 0 & 0 & 0 \\ 77 & 10 & 29 \\ 0 & 77 & 10 \\ 0 & 0 & 77 \\ 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad \mathcal{K}_{\mathbf{t},\beta}(\mathbf{Z},\mathbf{F}) = \begin{bmatrix} 50 & 58 & 0 \\ 0 & 50 & 58 \\ 0 & 0 & 50 \\ 77 & 10 & 29 \\ 27 & 49 & 29 \\ 0 & 0 & 0 \\ 0 & 77 & 10 \\ 0 & 27 & 49 \\ 0 & 0 & 77 \\ 0 & 0 & 27 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Besides, one can check that the shifted expansions of the relations $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$, and $\mathbf{p}_4$ with respect to $\mathbf{s}$ and $\mathbf{t}$ are

$$\begin{array}{rcl}
\mathcal{E}_{\mathbf{s},\beta}(\mathbf{p}_1) & = & [\ 96 \ \ 0 \ \ 0 \ \ 0 \ \ 96 \ \ 0 \ \ 0 \ \ 0 \ \ 1 \ \ 0 \ \ 0 \ \ 0\ ] \\
\mathcal{E}_{\mathbf{t},\beta}(\mathbf{p}_1) & = & [\ 96 \ \ 0 \ \ 0 \ \ 1 \ \ 96 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0\ ] \\
\mathcal{E}_{\mathbf{s},\beta}(\mathbf{p}_2) & = & [\ 13 \ \ 3 \ \ 0 \ \ 0 \ \ 57 \ \ 1 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0\ ] \\
\mathcal{E}_{\mathbf{t},\beta}(\mathbf{p}_2) & = & [\ 57 \ \ 1 \ \ 0 \ \ 0 \ \ 13 \ \ 0 \ \ 0 \ \ 3 \ \ 0 \ \ 0 \ \ 0 \ \ 0\ ] \\
\mathcal{E}_{\mathbf{s},\beta}(\mathbf{p}_3) & = & [\ 82 \ \ 40 \ \ 1 \ \ 0 \ \ 76 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0\ ] \\
\mathcal{E}_{\mathbf{t},\beta}(\mathbf{p}_3) & = & [\ 76 \ \ 0 \ \ 0 \ \ 0 \ \ 82 \ \ 0 \ \ 0 \ \ 40 \ \ 0 \ \ 1 \ \ 0 \ \ 0\ ] \\
\mathcal{E}_{\mathbf{s},\beta}(\mathbf{p}_4) & = & [\ 0 \ \ 0 \ \ 0 \ \ 1 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0\ ] \\
\mathcal{E}_{\mathbf{t},\beta}(\mathbf{p}_4) & = & [\ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 1\ ].
\end{array}$$

✍

4.2 Fast computation of the monomial basis

Let $\mathbf{M}$ be a tuple of r pairwise commuting matrices in $\mathbb{K}^{D\times D}$, let $\mathbf{F} \in \mathbb{K}^{m\times D}$, and let $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^{1\times m}$. Our goal is to solve Problem 2, that is, find a $\prec$-Gröbner

basis of relations of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$. In this section, we first show that the $\prec$-monomial basis of the quotient $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ corresponds to the row rank profile of the multi-Krylov matrix. Then, we give some properties about the structure of this matrix, and finally we exploit this structure to design an algorithm to compute this row rank profile efficiently.

4.2.1 Row rank profile and monomial basis

From our discussion about the left nullspace of the multi-Krylov matrix, we know that we will only be able to consider relations of bounded degree. Thus, although we have defined the multi-Krylov matrix for an arbitrary tuple $\boldsymbol{\beta}$, in the context of solving Problem 2 we must make sure that these bounds are chosen sufficiently large so that they allow us to retrieve a whole generating set of relations.

To be more precise, our aim is to compute the $\prec$-reduced Gröbner basis $\mathcal{G}$ for $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$. Therefore, we are particularly interested in the case where these integers are a priori upper bounds on the degrees of the generators in $\mathcal{G}$, which means that $\boldsymbol{\beta}$ is large enough so that the nullspace of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ contains the expansions of all the polynomials in $\mathcal{G}$. Formally, we will often consider the following assumption on $\boldsymbol{\beta}$.

Definition 4.9 (Assumption $\mathcal{H}_{\boldsymbol{\beta}}$)**.** *Let $\{\mathbf{p}_1,\dots,\mathbf{p}_s\}\in\mathbb{K}[\boldsymbol{X}]^{1\times m}$ be the $\prec$-reduced Gröbner relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$. Then, the assumption $\mathcal{H}_{\boldsymbol{\beta}}$ on $\boldsymbol{\beta}=(\beta_1,\dots,\beta_r)\in\mathbb{Z}^r_{>0}$ is the assertion that $\beta_k>\max_{1\leqslant j\leqslant s}\deg_{X_k}(\mathbf{p}_j)$ for $1\leqslant k\leqslant r$.*

One may have context-specific such bounds $\boldsymbol{\beta}$; if not, one can take for β_k a bound on the degree of the minimal polynomial of the multiplication matrix $\mathbf{M}_k$. In particular, $\boldsymbol{\beta}=(D+1,\dots,D+1)$ is always a valid choice.

We recall that, for a matrix $\mathbf{A}\in\mathbb{K}^{\mu\times\nu}$, the *row rank profile* of $\mathbf{A}$ is the lexicographically smallest subsequence $(\rho_1,\dots,\rho_\Delta)$ of $(1,\dots,\mu)$ such that $\Delta=\mathrm{rank}(\mathbf{A})$ and the rows $(\rho_1,\dots,\rho_\Delta)$ of $\mathbf{A}$ have rank Δ.

Theorem 4.10. *Let $\mathbf{M}=(\mathbf{M}_1,\dots,\mathbf{M}_r)$ be pairwise commuting matrices in $\mathbb{K}^{D\times D}$, let $\mathbf{F}\in\mathbb{K}^{m\times D}$, let $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^{1\times m}$, and let $\boldsymbol{\beta}=(\beta_1,\dots,\beta_r)\in\mathbb{Z}^r_{>0}$ be such that $\mathcal{H}_{\boldsymbol{\beta}}$. Let further $(\rho_1,\dots,\rho_\Delta)\in\mathbb{Z}^\Delta_{>0}$ be the row rank profile of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ and write $\rho_j=\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j})$ for $1\leqslant j\leqslant\Delta$. Then, $\{\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j},1\leqslant j\leqslant\Delta\}$ is the $\prec$-monomial basis of $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.*

Proof. We want to prove that the $\prec$-initial module of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ is the set of monomials not in $\{\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j},1\leqslant j\leqslant\Delta\}$.

First, consider any monomial $\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i$ for $1\leqslant i\leqslant m$ and $\mathbf{e}\in\mathbb{Z}^r_{\geqslant 0}$ such that $\mathbf{e}\not\prec\boldsymbol{\beta}$. Such a monomial cannot be in $\{\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j},1\leqslant j\leqslant\Delta\}$ since by construction of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ we have $\mathbf{e}_j<\boldsymbol{\beta}$ for all j. On the other hand, from the assumption $\mathcal{H}_{\boldsymbol{\beta}}$, $\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i$ cannot either be in the $\prec$-monomial basis of $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$, and thus $\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i\in\mathrm{in}_\prec(\mathrm{Syz}_{\mathbf{M}}(\mathbf{F}))$.

Now, let $\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i\in\mathrm{in}_\prec(\mathrm{Syz}_{\mathbf{M}}(\mathbf{F}))$ such that $\mathbf{e}<\boldsymbol{\beta}$. Then, there is a relation $\mathbf{p}\in\mathbb{K}[\boldsymbol{X}]^{1\times m}$ for $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ such that $\mathrm{in}_\prec(\mathbf{p})=\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i$, and Lemma 4.6 implies that $\mathcal{E}_{\prec,\boldsymbol{\beta}}(\mathbf{p})$ is in the left nullspace of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$. Furthermore, by construction, the rightmost nonzero entry of $\mathcal{E}_{\prec,\boldsymbol{\beta}}(\mathbf{p})$ is 1 at index $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i)$. In other words, $\mathcal{E}_{\prec,\boldsymbol{\beta}}(\mathbf{p})$ expresses the row of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ with index $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i)$ as a $\mathbb{K}$-linear combination of the rows with smaller

index. By definition of the row rank profile, this implies that $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i) \notin \{\rho_1, \ldots, \rho_\Delta\}$, and therefore $\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i \notin \{\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j}, 1 \leqslant j \leqslant \Delta\}$.

Conversely, let $\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i \notin \{\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j}, 1 \leqslant j \leqslant \Delta\}$ be a monomial such that $\mathbf{e} < \boldsymbol{\beta}$. Then, $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i) \notin \{\rho_1, \ldots, \rho_\Delta\}$. Thus, by definition of the row rank profile, there is a vector $\mathbf{v} \in \mathbb{K}^{1\times m\beta_1\cdots\beta_r}$ such that $\mathbf{v}$ is in the left nullspace of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M}, \mathbf{F})$ and the rightmost nonzero entry of $\mathbf{v}$ is 1 at index $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i)$. Then, $\mathrm{in}_\prec(\mathcal{C}_{\prec,\boldsymbol{\beta}}(\mathbf{v})) = \boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i$, and according to Lemma 4.6, $\mathcal{C}_{\prec,\boldsymbol{\beta}}(\mathbf{v})$ is a relation of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$, hence $\mathrm{in}_\prec(\mathcal{C}_{\prec,\boldsymbol{\beta}}(\mathbf{v})) \in \mathrm{in}_\prec(\mathrm{Syz}_{\mathbf{M}}(\mathbf{F}))$. ☙

As a direct consequence, we get the following refinement of Lemma 2.5 about the dimension of the quotient $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

Corollary 4.11. *Let $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$ be pairwise commuting matrices in $\mathbb{K}^{D\times D}$, let $\mathbf{F} \in \mathbb{K}^{m\times D}$, let $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^{1\times m}$, and let $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_r) \in \mathbb{Z}^r_{>0}$ be degree bounds such that $\mathcal{H}_{\boldsymbol{\beta}}$. Then, the dimension of $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ as a $\mathbb{K}$-vector space is equal to the rank of the multi-Krylov matrix $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M}, \mathbf{F})$.*

4.2.2 Structure and row rank profile of a multi-Krylov matrix

We note that the dense representation of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M}, \mathbf{F})$ uses $m\beta_1\cdots\beta_r D$ field elements. Taking the a priori bounds $\beta_1 = \cdots = \beta_r = D + 1$, this is in $\Theta(mD^{r+1})$, while our target cost bound $\tilde{\mathcal{O}}(mD^{\omega-1} + rD^\omega)$, stated precisely in Theorem 2.13, is sub-cubic in D. In fact, we will never compute the full dense representation of this matrix, like previous algorithms dealing with similar situations [MB82, KG85, FGLM93, MMM93]. Roughly, the idea is that once some monomial is found not to be in the monomial basis of the quotient, it can be discarded along with all its multiples.

The multi-Krylov matrix is succinctly described by $\mathbf{M}$, $\mathbf{F}$, $\prec$, and $\boldsymbol{\beta}$, whose representation uses $\mathcal{O}(mD + rD^2)$ field elements altogether. Yet, it is not straightforward how to deduce the row rank profile from this data. In the next section, we design an algorithm which uses in particular ideas from [KG85] for efficiency, and which only deals with submatrices of the multi-Krylov matrix that are represented using a number of field elements which remains within the bound above. (Hereafter, when we mention a *submatrix*, it is always assumed that the order of the rows in the original matrix is preserved.)

In this section, we discuss the property of structure of the multi-Krylov matrix that we exploit in this algorithm to avoid relying on its dense representation. When $\boldsymbol{\beta}$ satisfies $\mathcal{H}_{\boldsymbol{\beta}}$, this property can be summarized as being the mere translation in this linear algebra framework of the module structure of $\mathrm{in}_\prec(\mathrm{Syz}_{\mathbf{M}}(\mathbf{F}))$: once it is found that some monomial belongs to this initial module, then so do all its monomial multiples.

In the lemma below, we give details about this property, and we also give a similar property which holds without assuming $\mathcal{H}_{\boldsymbol{\beta}}$, and which still leads to an efficient algorithm to find the row rank profile of the multi-Krylov matrix. In linear algebra terms and in a nutshell, this structure indicates that we may discard a part of the rows that have not yet been computed and which correspond to right-multiples of rows that have already been found not to be in the row rank profile.

Lemma 4.12. *Let $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$ be pairwise commuting matrices in $\mathbb{K}^{D\times D}$, let $\mathbf{F} \in \mathbb{K}^{m\times D}$, let $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^{1\times m}$, let $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_r) \in \mathbb{Z}^r_{>0}$, and let*

$\boldsymbol{\rho} \in \mathbb{Z}_{>0}^{\Delta}$ denote the row rank profile of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$. Then, for any monomial $\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i$ in $\mathbb{K}[\boldsymbol{X}]^{1\times m}$ such that $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i)$ is not in $\boldsymbol{\rho}$, the following properties hold.

(i) If $\boldsymbol{\beta}$ satisfies the assumption $\mathcal{H}_{\boldsymbol{\beta}}$ of Definition 4.9, then $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}+\mathbf{e}'}\mathbf{c}_i)$ does not belong to $\boldsymbol{\rho}$ for any exponent $\mathbf{e}'$ such that $\mathbf{e}+\mathbf{e}' < \boldsymbol{\beta}$.

(ii) Let $\{(\mathbf{e}_j, i_j), 1 \leqslant j \leqslant s\}$ be such that $\mathbf{0} \leqslant \mathbf{e}_j < \boldsymbol{\beta}$, $1 \leqslant i_j \leqslant m$, and $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j}) < \phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i)$ for $1 \leqslant j \leqslant s$, and such that the row $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i)$ of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ is a linear combination of the rows $\{\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j}), 1 \leqslant j \leqslant s\}$. Then, $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}+\mathbf{e}'}\mathbf{c}_i)$ does not belong to $\boldsymbol{\rho}$ for any $\mathbf{e}'$ such that $\mathbf{e}+\mathbf{e}' < \boldsymbol{\beta}$ and $\mathbf{e}_j+\mathbf{e}' < \boldsymbol{\beta}$ for $1 \leqslant j \leqslant s$.

Proof. The item (i) directly follows from Theorem 4.10: the assumption ensures that $\boldsymbol{\rho}$ corresponds to the $\prec$-monomial basis of $\mathrm{in}_{\prec}(\mathrm{Syz}_{\mathbf{M}}(\mathbf{F}))$. Thus, the monomial $\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i$ is in $\mathrm{in}_{\prec}(\mathrm{Syz}_{\mathbf{M}}(\mathbf{F}))$, and any multiple $\boldsymbol{X}^{\mathbf{e}+\mathbf{e}'}\mathbf{c}_i$ of it is in $\mathrm{in}_{\prec}(\mathrm{Syz}_{\mathbf{M}}(\mathbf{F}))$ as well. It follows that $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}+\mathbf{e}'}\mathbf{c}_i)$ does not belong to $\boldsymbol{\rho}$.

Concerning the item (ii), it essentially follows from the construction of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$, and in particular from the ordering of its rows according to the indexing function $\phi_{\prec,\boldsymbol{\beta}}$ which is increasing with respect to the monomial order $\prec$.

The row $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i)$ of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ is the row $\mathbf{F}_{i,*}\mathbf{M}^{\mathbf{e}}$, which is assumed to be a linear combination of the rows $\mathbf{F}_{i_j,*}\mathbf{M}^{\mathbf{e}_j}$ for $1 \leqslant j \leqslant s$, with $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j}) < \phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i)$. Then, having $\mathbf{e}+\mathbf{e}' < \boldsymbol{\beta}$ and $\mathbf{e}_j+\mathbf{e}' < \boldsymbol{\beta}$ implies that the rows $\mathbf{F}_{i,*}\mathbf{M}^{\mathbf{e}+\mathbf{e}'}$ and $\mathbf{F}_{i_j,*}\mathbf{M}^{\mathbf{e}_j+\mathbf{e}'}$ are among the rows of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$, for $1 \leqslant j \leqslant s$; since these are precisely the rows mentioned above after right-multiplication by $\mathbf{M}^{\mathbf{e}'}$, we have that $\mathbf{F}_{i,*}\mathbf{M}^{\mathbf{e}+\mathbf{e}'}$ is a linear combination of the rows $\mathbf{F}_{i_j,*}\mathbf{M}^{\mathbf{e}_j+\mathbf{e}'}$.

We have proved that the row $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}+\mathbf{e}'}\mathbf{c}_i)$ of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ is a linear combination of its rows $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}_j+\mathbf{e}'}\mathbf{c}_{i_j})$ for $1 \leqslant j \leqslant s$. Having $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j}) < \phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i)$ means that $\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j} \prec \boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i$; this implies $\boldsymbol{X}^{\mathbf{e}_j+\mathbf{e}'}\mathbf{c}_{i_j} \prec \boldsymbol{X}^{\mathbf{e}+\mathbf{e}'}\mathbf{c}_i$, hence $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}_j+\mathbf{e}'}\mathbf{c}_{i_j}) < \phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}+\mathbf{e}'}\mathbf{c}_i)$. As a result, the row $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}+\mathbf{e}'}\mathbf{c}_i)$ of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ is a linear combination of the rows with smaller index, hence the conclusion: $\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}+\mathbf{e}'}\mathbf{c}_i)$ does not belong to $\boldsymbol{\rho}$. ☛

4.2.3 Computing the row rank profile of a multi-Krylov matrix

Now, we give a fast algorithm to compute the row rank profile of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$. We exploit the structure of the matrix, and in particular the item (ii) of Lemma 4.12. For further efficiency, the algorithm below resorts to fast linear algebra techniques.

In particular, our algorithm relies on fast matrix multiplication by following a strategy in the style of Keller-Gehrig [KG85]. In short, this can be thought of as precomputing powers of the multiplication matrices of the form $\mathbf{M}_j^{2^e}$, which then allow us to group many vector-matrix products into fewer matrix-matrix products. In order to effectively group these products, we work iteratively on the variables; this way, we first focus on all operations involving $\mathbf{M}_1$, then those involving $\mathbf{M}_2$, etc.

We remark that the order of the rows specified by the monomial order $\prec$ is not respected in the process, since at a fixed stage of the algorithm we will only have considered a submatrix of the multi-Krylov matrix which does not involve any of the last variables. Yet, by constantly re-ordering, according to $\prec$, the rows that have been processed and the ones that we introduce, we manage to respect the monomial order in the end.

A key building block is a procedure RowRankProfile which computes the row rank profile of any matrix in $\mathbb{K}^{\mu\times\nu}$ of rank ρ in $\mathcal{O}(\rho^{\omega-2}\mu\nu)$ operations in $\mathbb{K}$. Such an algorithm can be found in [Sto00, Section 2.2], with a cost bound given in [Sto00, Theorem 2.10].

For simplicity of presentation, and since this is sufficient for our needs, in Algorithm 1 we assume that the degree bounds in $\boldsymbol{\beta}$ are powers of 2. If need be, one may easily adapt the algorithm to work with arbitrary bounds $\boldsymbol{\beta}$: one would simply have to discard some of the rows of $\mathbf{BP}$ at Step **7.b.**(vi) at the last iteration of the While loop for each variable.

Proposition 4.13. *Algorithm 1 is correct and uses*

$$\begin{aligned}&\mathcal{O}\big(\rho^{\omega-2}mD + D^{\omega}\log\big(\textstyle\prod_{1\leqslant k\leqslant r}\min(\beta_k,\Delta+1)\big)\big)\\ &\quad\subseteq \mathcal{O}\big(mD^{\omega-1} + rD^{\omega}\log(\Delta+1)\big)\end{aligned}$$

operations in $\mathbb{K}$, where ρ is the rank of $\mathbf{F}$ and Δ is the dimension of $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$ as a $\mathbb{K}$-vector space.

Proof. For $1\leqslant k\leqslant r$ and $0\leqslant e\leqslant \log_2(\beta_k)$, let us consider the set of monomials

$$\mathcal{S}_{k,e} = \{\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i,\ 1\leqslant i\leqslant m,\ 0\leqslant \mathbf{e} < (\beta_1,\dots,\beta_{k-1},2^e,1,\dots,1)\}.$$

Then, we denote by $\mathbf{C}_{k,e}\in\mathbb{K}^{m\beta_1\cdots\beta_{k-1}2^e\times D}$ the submatrix of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ formed by its rows in $\phi_{\prec,\boldsymbol{\beta}}(\mathcal{S}_{k,e})$. Notice that the redundancy $\mathbf{C}_{k,\log_2(\beta_k)} = \mathbf{C}_{k+1,0}$ is voluntarily introduced in order to simplify the exposition of the proof.

Then, $\mathbf{C}_{1,0}$ is the submatrix of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ with rows in $\{\phi_{\prec,\boldsymbol{\beta}}(\mathbf{c}_1),\dots,\phi_{\prec,\boldsymbol{\beta}}(\mathbf{c}_m)\}$; therefore, by choice of the permutation π at Step **2**, we have $\mathbf{C}_{1,0}=\pi\mathbf{F}$. Thus, after performing Step **6**, the matrix $\mathbf{B}$ is formed by the rows corresponding to the row rank profile of $\mathbf{C}_{1,0}$, and $(\rho_1,\dots,\rho_\delta)$ are the indices of these rows in $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$.

Then, let $k\in\{1,\dots,r\}$ and $0\leqslant e<\log_2(\beta_k)$, and assume that $\mathbf{B}$ is formed by the rows corresponding to the row rank profile of $\mathbf{C}_{k,e}$, and that $(\rho_1,\dots,\rho_\delta)$ are the indices of these rows in $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$. (We just proved that this holds when $k=1$ and $e=0$.) We place ourselves at the beginning of the iteration (k,e) of the For and While loops, and we are going to prove that this iteration preserves the above property; this will directly imply the correctness of the algorithm, which precisely asks that this property holds for $k=r$ and $e=\log_2(\beta_r)$.

Let us denote $\boldsymbol{\rho}=\{\rho_1,\dots,\rho_\delta\}$ and $\hat{\boldsymbol{\rho}}=\{\hat\rho_1,\dots,\hat\rho_\delta\}$, where $\hat\rho_j=\phi_{\prec,\boldsymbol{\beta}}(X_k^{2^e}\phi^{-1}_{\prec,\boldsymbol{\beta}}(\rho_j))$ for $1\leqslant j\leqslant\delta$ are the indices computed at Step **7.b.**(i). Let also $(\gamma_1,\dots,\gamma_\nu)$ be the indices of the rows of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ corresponding to the row rank profile of its submatrix $\mathbf{C}_{k,e+1}$. Then, we claim that:

- *Fact 1:* $(\gamma_1,\dots,\gamma_\nu)$ forms a subsequence of the tuple $\mathbf{t}$, or in other words, $\{\gamma_1,\dots,\gamma_\nu\}$ is a subset of $\boldsymbol{\rho}\cup\hat{\boldsymbol{\rho}}$;
- *Fact 2:* if we have equality $\{\gamma_1,\dots,\gamma_\nu\}=\boldsymbol{\rho}$, then $\boldsymbol{\rho}$ is the set of indices of the rows of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ corresponding to the row rank profile of $\mathbf{C}_{k,\log_2(\beta_k)}$.

In particular, *Fact 2* explains why, when break is set to True then the While loop can be exited, and if $k<r$ then we can turn to the next variable X_{k+1}, noting that the transition is ensured by $\mathbf{C}_{k,\log_2(\beta_k)}=\mathbf{C}_{k+1,0}$.

Algorithm 1 – KrylovRankProf
(Row rank profile of a multi-Krylov matrix)

Input:

- pairwise commuting matrices $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$ in $\mathbb{K}^{D\times D}$,
- matrix $\mathbf{F} \in \mathbb{K}^{m\times D}$,
- monomial order $\prec$ on $\mathbb{K}[X_1, \ldots, X_r]^{1\times m}$,
- degree bounds $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_r) \in \mathbb{Z}^r_{>0}$ which are powers of 2.

Output:

- the row rank profile of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M}, \mathbf{F})$,
- the submatrix of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M}, \mathbf{F})$ formed by the corresponding rows.

1. $\phi_{\prec,\boldsymbol{\beta}} \leftarrow$ the indexing function in Definition 4.2

2. $\pi \leftarrow$ the permutation matrix in $\{0,1\}^{m\times m}$ such that the tuple $\mathbf{t} = \pi[\phi_{\prec,\boldsymbol{\beta}}(\mathbf{c}_1), \ldots, \phi_{\prec,\boldsymbol{\beta}}(\mathbf{c}_m)]^\mathsf{T}$ is increasing

3. $\mathbf{B} \leftarrow \pi\mathbf{F}$

4. $\delta, (i_1, \ldots, i_\delta) \leftarrow$ RowRankProfile($\mathbf{B}$)

5. $(\rho_1, \ldots, \rho_\delta) \leftarrow$ the subtuple of $\mathbf{t}$ with its entries $(i_1, \ldots, i_\delta)$

6. $\mathbf{B} \leftarrow$ the submatrix of $\mathbf{B}$ with its rows $(i_1, \ldots, i_\delta)$

7. For k from 1 to r `// iterate over the variables`

- **a.** $\mathbf{P} \leftarrow \mathbf{M}_k$; $e \leftarrow 0$; break $\leftarrow$ False
- **b.** While $e < \log_2(\beta_k)$ and (not break)
  - (i) $\hat{\rho}_j \leftarrow \phi_{\prec,\boldsymbol{\beta}}(X_k^{2^e}\phi^{-1}_{\prec,\boldsymbol{\beta}}(\rho_j))$ for $1 \leqslant j \leqslant \delta$
  - (ii) $\pi \leftarrow$ permutation matrix in $\{0,1\}^{2\delta\times 2\delta}$ such that the tuple $\mathbf{t} = \pi[\rho_1, \ldots, \rho_\delta, \hat{\rho}_1, \ldots, \hat{\rho}_\delta]^\mathsf{T}$ is increasing
  - (iii) $\mathbf{B} \leftarrow \pi \begin{bmatrix} \mathbf{B} \\ \mathbf{BP} \end{bmatrix}$
  - (iv) $\hat{\delta}, (i_1, \ldots, i_{\hat{\delta}}) \leftarrow$ RowRankProfile($\mathbf{B}$)
  - (v) If $(\rho_1, \ldots, \rho_\delta) =$ subtuple of $\mathbf{t}$ with its entries $(i_1, \ldots, i_{\hat{\delta}})$
    - break $\leftarrow$ True
  - (vi) Else
    - $(\rho_1, \ldots, \rho_{\hat{\delta}}) \leftarrow$ subtuple of $\mathbf{t}$ with its entries $(i_1, \ldots, i_{\hat{\delta}})$
    - $\mathbf{B} \leftarrow$ submatrix of $\mathbf{B}$ with its rows $(i_1, \ldots, i_{\hat{\delta}})$
    - $\delta \leftarrow \hat{\delta}$; $\mathbf{P} \leftarrow \mathbf{P}^2$; $e \leftarrow e + 1$

8. Return $(\rho_1, \ldots, \rho_\delta)$ and $\mathbf{B}$

Proof of *Fact 1*. Let $1 \leqslant j \leqslant \nu$ and let us prove that $\gamma_j \in \boldsymbol{\rho} \cup \hat{\boldsymbol{\rho}}$. By assumption, $\boldsymbol{\rho}$ are the indices of the rows in $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ corresponding to the row rank profile of its submatrix $\mathbf{C}_{k,e}$. Then, the item (ii) of Lemma 4.12 implies that the monomial $\phi^{-1}_{\prec,\boldsymbol{\beta}}(\gamma_j)$ is not a multiple of a monomial in $\phi^{-1}_{\prec,\boldsymbol{\beta}}(\phi_{\prec,\boldsymbol{\beta}}(\mathcal{S}_{k,e}) - \boldsymbol{\rho}) = \mathcal{S}_{k,e} - \phi^{-1}_{\prec,\boldsymbol{\beta}}(\boldsymbol{\rho})$.

If $\phi^{-1}_{\prec,\boldsymbol{\beta}}(\gamma_j) \in \mathcal{S}_{k,e}$, this implies $\gamma_j \in \boldsymbol{\rho}$. Now, we assume that $\phi^{-1}_{\prec,\boldsymbol{\beta}}(\gamma_j) \in \mathcal{S}_{k,e+1} - \mathcal{S}_{k,e}$, and we prove that $\gamma_j \in \hat{\boldsymbol{\rho}}$, or in other words, that $\phi^{-1}_{\prec,\boldsymbol{\beta}}(\gamma_j) \in \{X_k^{2^e}\phi^{-1}_{\prec,\boldsymbol{\beta}}(\rho_j), 1 \leqslant j \leqslant \delta\}$. Since $\phi^{-1}_{\prec,\boldsymbol{\beta}}(\gamma_j) \in \mathcal{S}_{k,e+1} - \mathcal{S}_{k,e}$, we can write $\phi^{-1}_{\prec,\boldsymbol{\beta}}(\gamma_j) = X_k^{2^e} f$ for some $f \in \mathcal{S}_{k,e}$. Then, $X_k^{2^e} f$ not being a multiple of a monomial in $\mathcal{S}_{k,e} - \phi^{-1}_{\prec,\boldsymbol{\beta}}(\boldsymbol{\rho})$ implies that $f \in \phi^{-1}_{\prec,\boldsymbol{\beta}}(\boldsymbol{\rho})$, hence the conclusion.

Proof of *Fact 2*. Any monomial in $\mathcal{S}_{k,\log_2(\beta_k)} - \mathcal{S}_{k,e}$ is a multiple of a monomial in $\mathcal{S}_{k,e+1} - \mathcal{S}_{k,e}$. On the other hand, by assumption, all monomials corresponding to the row rank profile of $\mathbf{C}_{k,e+1}$ are in $\mathcal{S}_{k,e}$. Therefore $\mathcal{S}_{k,e+1} - \mathcal{S}_{k,e}$ only contains rows of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ which are not in its row rank profile, and the conclusion follows from Lemma 4.12.

Then, knowing that $(\gamma_1, \ldots, \gamma_\nu)$ is a subsequence of $\mathbf{t}$, at the end of the iteration (k, e) we have that $\mathbf{B}$ is formed by the rows corresponding to the row rank profile of $\mathbf{C}_{k,e+1}$, and $(\rho_1, \ldots, \rho_\delta)$ are the indices of these rows in $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$. Furthermore, if the variable break has been set to True during this iteration, then $\mathbf{B}$ corresponds to the row rank profile of $\mathbf{C}_{k,\log_2(\beta_k)}$. This concludes the proof of correctness.

Concerning the cost bound, according to [Sto00, Theorem 2.10], the row rank profile computation at Step **4** can be done in $\mathcal{O}(\rho^{\omega-2}mD)$ operations, where $\rho = \operatorname{rank}(\mathbf{F})$.

Let us now focus on the iteration (k, e) and show that it uses in $\mathcal{O}(D^\omega)$ operations. First, the computation of $\mathbf{BP}$ at Step **7.b.**(iii), where the matrix $\mathbf{P}$ has dimensions $D \times D$ and $\mathbf{B}$ has D columns and $\delta \leqslant D$ rows, can be performed in $\mathcal{O}(D^\omega)$ operations. Then, since $\mathbf{B}$ at Step **7.b.**(iv) has $2\delta \leqslant 2D$ rows and D columns, its row rank profile can be computed in $\mathcal{O}(D^\omega)$ operations. Finally, squaring the $D \times D$ matrix $\mathbf{P}$ at Step **7.b.**(vi) is also done in $\mathcal{O}(D^\omega)$ operations.

To conclude the proof of the cost bound, we claim that in the iteration k of the For loop, the number of iterations of the While loop cannot exceed $1 + \log_2(\Delta + 1)$. From Theorem 4.10, we know that the rank of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ is at most Δ. Then, once we introduce rows of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ involving powers of the variable X_k greater than Δ, that is, in iteration $e > \log_2(\Delta)$ of the While loop, then these rows are linear combinations of the previously introduced rows with $0 \leqslant e \leqslant \log_2(\Delta)$. As a result, break will be set to true and the While loop is exited. ∎

We note that in most interesting cases we have $\Delta > 1$ and $m \in \mathcal{O}(D)$. This implies that $\log(\Delta + 1) = \Theta(\log(\Delta))$, and that the first term $\mathcal{O}(\rho^{\omega-2}mD)$ in the cost bound is in $\mathcal{O}(D^\omega)$ and is thus dominated by the second term. In this case, the cost bound can be simplified as $\mathcal{O}(rD^\omega \log(\Delta))$.

Combining this algorithm with Theorem 4.10, we obtain the following result.

Corollary 4.14. *Let $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$ be pairwise commuting matrices in $\mathbb{K}^{D\times D}$, let $\mathbf{F} \in \mathbb{K}^{m\times D}$, and let $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^{1\times m}$. Let further $\boldsymbol{\beta} \in \mathbb{Z}_{>0}$ be a priori degree bounds such that $\mathcal{H}_{\boldsymbol{\beta}}$ and $\boldsymbol{\beta} \leqslant (D+1, \ldots, D+1)$. Then, there is an algorithm*

which computes the $\prec$-monomial basis of $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$ using

$$\begin{aligned}&\mathcal{O}\big(\rho^{\omega-2}mD + D^{\omega}\log\big(\textstyle\prod_{1\leqslant k\leqslant r}\min(\beta_k,\Delta+1)\big)\big)\\ &\subseteq \mathcal{O}(mD^{\omega-1} + rD^{\omega}\log(\Delta+1))\end{aligned}$$

operations in $\mathbb{K}$, where ρ is the rank of $\mathbf{F}$ and Δ is the dimension of $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$ as a $\mathbb{K}$-vector space.

Proof. According to Theorem 4.10, the $\prec$-monomial basis can directly be computed by Algorithm 1, up to the choice of degree bounds that satisfy $\mathcal{H}_{\boldsymbol{\beta}}$ and which are powers of 2. By default, since $(D+1,\dots,D+1)$ satisfies $\mathcal{H}_{\boldsymbol{\beta}}$, one may for example take the tuple $\boldsymbol{\beta} = (2^{\lceil\log_2(D+1)\rceil},\dots,2^{\lceil\log_2(D+1)\rceil})$. The cost bound follows from Proposition 4.13. ☙

Remark 4.15. The order in which the For and While loops introduce the new monomials to be processed precisely corresponds to the $\prec_{\text{lex}}$-term over position order $\prec_{\text{lex}}^{\text{top}}$ over $\mathbb{K}[\boldsymbol{X}]^{1\times m}$. As a result, the behaviour and the cost bound of the algorithm can be described with more precision if the input monomial order is $\prec = \prec_{\text{lex}}^{\text{top}}$.

In this case, we are processing the rows of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ in the order they are in the matrix. In particular, the permutation π at Steps **2** and **7.b.**(ii) is always the identity matrix, and the tuple $(\rho_1,\dots,\rho_\delta)$ at Step **5** or inside the loops consists of the first δ entries of the actual row rank profile of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$.

Furthermore, the fact that we are processing the rows in their actual order has a small impact on the cost bound, as follows. Let us denote by $\mathcal{G}$ the $\prec$-reduced Gröbner relation basis for $\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$, and let $\hat{\boldsymbol{\beta}} = (\hat\beta_1,\dots,\hat\beta_r)$ be the tuple of maximum degrees in $\mathcal{G}$, that is, $\hat\beta_k = \max_{\mathbf{p}\in\mathcal{G}}\deg_{X_k}(\mathbf{p})$ for $1\leqslant k\leqslant r$. We assume $\mathcal{H}_{\boldsymbol{\beta}}$, which states that $\hat{\boldsymbol{\beta}} < \boldsymbol{\beta}$.

Then, at the iteration k of the For loop, the While loop does $\mathcal{O}(\log_2(\hat\beta_k+1))$ iterations. Indeed, once we introduce powers of the variable X_k greater than $\hat\beta_k$, the partial row rank profile $(\rho_1,\dots,\rho_\delta)$ will not be modified anymore, and break will be set to true. Therefore the total number of iterations is $\mathcal{O}(\sum_{1\leqslant k\leqslant r}\log(\hat\beta_k+1))$.

The arithmetic mean-geometric mean inequality gives

$$\sum_{1\leqslant k\leqslant r}\log(\hat\beta_k+1) \;\leqslant\; r\log\left(\frac{\hat\beta_1+\cdots+\hat\beta_r}{r}+1\right).$$

Besides we have $\hat\beta_1+\cdots+\hat\beta_r \leqslant \Delta+r-1$ since the $\prec$-monomial basis, of cardinality Δ, contains the $1+\hat\beta_1+\cdots+\hat\beta_r-r$ distinct elements $\{1\}\cup\{X_k^e, 1\leqslant e<\hat\beta_k, 1\leqslant k\leqslant r\}$. Then, the number of iterations can be bounded as $\mathcal{O}(\sum_{1\leqslant k\leqslant r}\log(\hat\beta_k+1)) \subseteq \mathcal{O}(r\log(2+\Delta/r))$.

To summarize, when the input order is $\prec = \prec_{\text{lex}}^{\text{top}}$, and under the assumption $\mathcal{H}_{\boldsymbol{\beta}}$, Algorithm 1 uses

$$\mathcal{O}\left(\rho^{\omega-2}mD + rD^{\omega}\log\left(\frac{\Delta}{r}+2\right)\right)$$

operations in $\mathbb{K}$. ☕

4.3 Fast computation of the relation basis

In this section, we present our fast algorithm to compute the reduced Gröbner relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$. By definition, it can be described by the minimal generators of the initial module of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ along with the associated normal forms. We first show how to use the knowledge of the monomial basis to compute such normal forms efficiently.

4.3.1 Simultaneous computation of normal forms of monomials

Let $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$ be pairwise commuting matrices in $\mathbb{K}^{D\times D}$, let $\mathbf{F} \in \mathbb{K}^{m\times D}$, and let $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^{1\times m}$. Given some monomials $\{\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j}, 1 \leqslant j \leqslant s\}$, we want to compute their $\prec$-normal forms with respect to the module $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$. More precisely, for $1 \leqslant j \leqslant s$, the monomial $\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j}$ considered modulo $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ can be uniquely written as a $\mathbb{K}$-linear combination of the monomials in the $\prec$-monomial basis of $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$; we will focus on computing the coefficients of this combination.

In the linearized viewpoint, these monomials correspond to a matrix $\mathbf{T} \in \mathbb{K}^{s\times D}$, whose row j is $\mathbf{F}_{i_j,*}\mathbf{M}^{\mathbf{e}_j}$. Similarly, the monomials in the $\prec$-monomial basis of $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ form the rows of a matrix $\mathbf{B} \in \mathbb{K}^{\Delta\times D}$. According to Theorem 4.10, if we are given degree bounds $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_r) \in \mathbb{Z}^r_{>0}$ such that $\mathcal{H}_{\boldsymbol{\beta}}$, then $\mathbf{B}$ is the submatrix of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M}, \mathbf{F})$ formed by the rows in its row rank profile $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_\Delta) \in \mathbb{Z}^\Delta_{>0}$.

Then, the rows of $\mathbf{T}$ are $\mathbb{K}$-linear combinations of the rows of $\mathbf{B}$; these combinations may be gathered in a matrix $\mathbf{N} \in \mathbb{K}^{s\times\Delta}$ such that $\mathbf{T} = \mathbf{N}\mathbf{B}$. (The notation $\mathbf{T}$ stands for *terms*, while $\mathbf{B}$ stands for *basis*, and $\mathbf{N}$ for *normal forms*.) Our goal is to find this matrix $\mathbf{N}$; here, the fact that it is unique can be seen from the fact that $\mathbf{B}$ has full row rank.

Explicitly, $\mathbf{N}$ can be computed as follows from $\mathbf{T}$ and $\mathbf{B}$. If $\mathbf{B}$ is square, then we directly have $\mathbf{N} = \mathbf{T}\mathbf{B}^{-1}$. More generally, let $(\hat{\rho}_1, \ldots, \hat{\rho}_\Delta)$ denote the column rank profile of $\mathbf{B}$, and let $\hat{\mathbf{B}} \in \mathbb{K}^{\Delta\times\Delta}$ and $\hat{\mathbf{T}} \in \mathbb{K}^{s\times\Delta}$ be the submatrices of $\mathbf{B}$ and $\mathbf{T}$ formed by their columns $\{\hat{\rho}_1, \ldots, \hat{\rho}_\Delta\}$. Then, we have $\hat{\mathbf{T}} = \mathbf{N}\hat{\mathbf{B}}$ and $\hat{\mathbf{B}}$ is invertible, so that $\mathbf{N} = \hat{\mathbf{T}}\hat{\mathbf{B}}^{-1}$.

Finally, to obtain the sought normal forms it remains to compress back the linear relations given by the rows of $\mathbf{N}$ into polynomials in $\mathbb{K}[\boldsymbol{X}]^{1\times m}$. By construction, $\mathbf{N}$ is seen as the submatrix formed by the columns $(\rho_1, \ldots, \rho_\Delta)$ of a matrix $\hat{\mathbf{N}} \in \mathbb{K}^{s\times\beta_1\cdots\beta_r m}$, whose other columns are zero. Then, the $\prec$-normal forms of $\{\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j}, 1 \leqslant j \leqslant s\}$ are obtained as the compressions $\mathcal{C}_{\prec,\boldsymbol{\beta}}(\hat{\mathbf{N}}_{j,*})$ for $1 \leqslant j \leqslant s$.

Proposition 4.16. *Algorithm 2 is correct and uses*

$$\mathcal{O}(\Delta^{\omega-1}(D+s))$$

operations in $\mathbb{K}$.

Proof. The correctness follows from the discussion above. Concerning the cost bound, we first remark that Steps **2**, **3**, **5**, **6**, **7** do not use field operations.

Then, Step **1** uses $\mathcal{O}(\Delta^{\omega-1}D)$ field operations according to [Sto00, Theorem 2.10]. At Step **4**, the inversion of $\hat{\mathbf{B}}$ uses $\mathcal{O}(\Delta^\omega)$ operations. Then, the multiplication $\hat{\mathbf{T}}\hat{\mathbf{B}}^{-1}$ uses $\mathcal{O}(s\Delta^{\omega-1})$ operations if $s \geqslant \Delta$, and $\mathcal{O}(\Delta^\omega)$ otherwise. Since $\Delta \leqslant D$ we obtain the announced bound. ∎

Algorithm 2 – LinNormalForm
(Normal forms via linear algebra)

Input:

- pairwise commuting matrices $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$ in $\mathbb{K}^{D\times D}$,
- matrix $\mathbf{F} \in \mathbb{K}^{m\times D}$,
- matrix $\mathbf{T} \in \mathbb{K}^{s\times D}$ whose row j is $\mathbf{F}_{i_j,*}\mathbf{M}^{\mathbf{e}_j}$ for $1 \leqslant j \leqslant s$,
- monomial order $\prec$ on $\mathbb{K}[X_1, \ldots, X_r]^{1\times m}$,
- degree bounds $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_r) \in \mathbb{Z}^r_{>0}$ such that $\mathcal{H}_{\boldsymbol{\beta}}$,
- submatrix $\mathbf{B} \in \mathbb{K}^{\Delta\times D}$ of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M}, \mathbf{F})$ formed by its rows corresponding to its row rank profile $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_\Delta) \in \mathbb{Z}^\Delta_{>0}$.

Output: matrix in $\mathbb{K}[X_1, \ldots, X_r]^{s\times m}$ such that, for $1 \leqslant j \leqslant s$, its row j is the $\prec$-normal form $\mathrm{nf}_\prec(\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j})$ with respect to $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

1. $(\hat\rho_1, \ldots, \hat\rho_\Delta) \leftarrow$ the column rank profile of $\mathbf{B}$
2. $\hat{\mathbf{B}} \leftarrow$ submatrix of $\mathbf{B}$ formed by its columns $\{\hat\rho_1, \ldots, \hat\rho_\Delta\}$
3. $\hat{\mathbf{T}} \leftarrow$ submatrix of $\mathbf{T}$ formed by its columns $\{\hat\rho_1, \ldots, \hat\rho_\Delta\}$
4. $\mathbf{N} \leftarrow \hat{\mathbf{T}}\hat{\mathbf{B}}^{-1}$
5. $\hat{\mathbf{N}} \leftarrow$ zero matrix in $\mathbb{K}^{s\times D}$
6. $\hat{\mathbf{N}}_{*,\rho_k} \leftarrow \mathbf{N}_{*,k}$ for k from 1 to Δ
7. Return the matrix in $\mathbb{K}[X_1, \ldots, X_r]^{s\times m}$ whose row j is $\mathcal{C}_{\prec,\boldsymbol{\beta}}(\hat{\mathbf{N}}_{j,*})$

Remark 4.17. It is in fact not necessary to assume that the degree bounds $\boldsymbol{\beta}$ satisfy the assumption $\mathcal{H}_{\boldsymbol{\beta}}$ in order to perform normal form computation with Algorithm 2. Indeed, all we need is that the $\prec$-normal forms of the input monomials be $\mathbb{K}$-linear combinations of the monomials in $\phi^{-1}_{\prec,\boldsymbol{\beta}}(\boldsymbol{\rho})$.

For example, by definition of the row rank profile, this is the case if the input monomials all have degree bounded by $\boldsymbol{\beta}$, or in other words, if $\mathbf{T}$ is the submatrix of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M},\mathbf{F})$ formed by its rows $\{\phi_{\prec,\boldsymbol{\beta}}(\boldsymbol{X}^{\mathbf{e}_j}\mathbf{c}_{i_j}), 1 \leqslant j \leqslant s\}$. Still, in what follows, $\mathcal{H}_{\boldsymbol{\beta}}$ will be satisfied, since our goal is to rely on this normal form computation to compute the $\prec$-reduced Gröbner basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$. ☕

4.3.2 Univariate case: computing shifted Popov relation bases

To introduce our fast relation basis algorithm, we first focus on the univariate case. Let $\mathbf{M} \in \mathbb{K}^{D\times D}$, $\mathbf{F} \in \mathbb{K}^{m\times D}$, and $\mathbf{s} \in \mathbb{Z}^m$. Previous work using such a linear algebra viewpoint to compute Popov and Hermite bases includes [Kai80, Vil96].

Then, let $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m) \in \mathbb{Z}^m_{>0}$ be the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$. We have seen in Section 1.1.3 and Example 1.40 that the $<^{\mathbf{s}\text{-top}}$-monomial basis of $\mathbb{K}[X]^{1\times m}/\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ is given by $\boldsymbol{\delta}$ as $\cup_{1\leqslant i\leqslant m}\{X^e\mathbf{c}_i, 0 \leqslant e < \delta_i\}$. We also recall that, as a consequence, the dimension of $\mathbb{K}[X]^{1\times m}/\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ as a $\mathbb{K}$-vector space is $\Delta = \delta_1 + \cdots + \delta_m$.

Furthermore, from Section 4.2, we know that this monomial basis can be obtained from the row rank profile of the multi-Krylov matrix. More precisely, suppose that we have an a priori bound $\beta \in \mathbb{Z}_{>0}$ on the degree of the $\mathbf{s}$-Popov relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$, that is, $\beta > \max(\boldsymbol{\delta})$. Then, let $(\rho_1, \ldots, \rho_\Delta) \in \mathbb{Z}^\Delta_{>0}$ be the row rank profile of $\mathcal{K}_{\mathbf{s},\beta}(\mathbf{M},\mathbf{F})$ and write $\rho_j = \phi_{\mathbf{s},\beta}(X^{e_j}\mathbf{c}_{i_j})$ for $1 \leqslant j \leqslant \Delta$. Theorem 4.10 states that the $<^{\mathbf{s}\text{-top}}$-monomial basis of $\mathbb{K}[X]^{1\times m}/\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ is $\{X^{e_j}\mathbf{c}_{i_j}, 1 \leqslant j \leqslant \Delta\}$.

As a consequence, the $\mathbf{s}$-minimal degree $\boldsymbol{\delta}$ can be obtained as

$$\delta_i = \begin{cases} 1 + \max\{e_j \mid 1 \leqslant j \leqslant \Delta \text{ and } i_j = i\} & \text{if the set is nonempty,} \\ 0 & \text{if it is empty.} \end{cases}$$

Then, knowing $\boldsymbol{\delta}$, the sought $\mathbf{s}$-Popov relation basis can be computed as the matrix whose row i is $X^{\delta_i}\mathbf{c}_i - \mathrm{nf}_{<^{\mathbf{s}\text{-top}}}(X^{\delta_i}\mathbf{c}_i)$, as explained in Section 1.3.5.

These m normal forms can be computed efficiently using Algorithm 2, up to the preliminary computation of the submatrix $\mathbf{T}$ of $\mathcal{K}_{\mathbf{s},\beta}(\mathbf{M},\mathbf{F})$ formed by the rows with indices in $\{\phi_{\mathbf{s},\beta}(X^{\delta_1}\mathbf{c}_1), \ldots, \phi_{\mathbf{s},\beta}(X^{\delta_m}\mathbf{c}_m)\}$.

Proposition 4.18. *Algorithm 3 is correct and uses*

$$\mathcal{O}(mD^{\omega-1} + D^\omega \log(\min(\beta,\Delta))) \subseteq \mathcal{O}(mD^{\omega-1} + D^\omega \log(D))$$

operations in $\mathbb{K}$*, where* $\Delta = \delta_1 + \cdots + \delta_m$ *is the sum of the* $\mathbf{s}$*-minimal degrees of* $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$*.*

Proof. The correctness of the algorithm follows from the discussion above.

According to Corollary 4.14, the computation of the row rank profile and of the submatrix $\mathbf{B}$ at Step **1** uses $\mathcal{O}(mD^{\omega-1}+D^\omega \log(\min(\beta,\Delta)))$ field operations. Then, at Step **2**, the submatrix of $\mathbf{T}$ of its rows i such that $\delta_i > 0$ has at most $\min(m,\Delta) \leqslant D$ rows since $\delta_1 + \cdots + \delta_m = \Delta$. Thus, right-multiplying this submatrix by the $D \times D$ matrix $\mathbf{M}$ can be done in $\mathcal{O}(D^\omega)$ operations. Finally, the normal forms at Step **3** are computed in $\mathcal{O}(\Delta^{\omega-1}(D+m)) \subseteq \mathcal{O}(mD^{\omega-1} + D^\omega)$ operations according to Proposition 4.16. 🐎

Algorithm 3 – LinPopovRelBas
(Shifted Popov relation bases via linear algebra)

Input:

- matrix $\mathbf{M} \in \mathbb{K}^{D \times D}$,
- matrix $\mathbf{F} \in \mathbb{K}^{m \times D}$,
- shift $\mathbf{s} \in \mathbb{Z}^m$,
- bound $\beta \in \mathbb{Z}_{>0}$ on the degree of the $\mathbf{s}$-Popov relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ (defaults to $\beta = D + 1$).

Output: the $\mathbf{s}$-Popov relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

1. `/* Compute monomial basis and multi-Krylov submatrix */`
   $<^{\mathbf{s}\text{-top}} \leftarrow$ monomial order on $\mathbb{K}[X]^{1 \times m}$ as in Example 1.38
   $\phi_{\mathbf{s},\beta} \leftarrow$ indexing function as in Definition 4.2 and Section 4.1.3
   $(\rho_1, \ldots, \rho_\Delta), \mathbf{B} \leftarrow \mathrm{KrylovRankProf}(\mathbf{M}, \mathbf{F}, <^{\mathbf{s}\text{-top}}, 2^{\lceil \log_2(\beta) \rceil})$
   $X^{e_j} \mathbf{c}_{i_j} \leftarrow \phi^{-1}_{\mathbf{s},\beta}(\rho_j)$ for j from 1 to Δ
2. `/* Compute linearization of the` m `leading monomials */`
   $\mathbf{T} \leftarrow$ zero matrix in $\mathbb{K}^{m \times D}$
   For i from 1 to m
   a. $\mathcal{S}_i \leftarrow \{e_j \mid 1 \leqslant j \leqslant \Delta \text{ and } i_j = i\}$
   b. If $\mathcal{S}_i = \emptyset$:
   (i) $\delta_i \leftarrow 0$
   (ii) $\mathbf{T}_{i,*} \leftarrow \mathbf{F}_{i,*}$
   c. Else:
   (i) $\delta_i \leftarrow 1 + \max(\mathcal{S}_i)$
   (ii) $\mathbf{T}_{i,*} \leftarrow \mathbf{B}_{\rho_j,*}$ where j is the integer such that $e_j = \max(\mathcal{S}_i)$
   Right-multiply by $\mathbf{M}$ the submatrix of $\mathbf{T}$ of its rows i such that $\delta_i > 0$
3. `/* Compute normal forms of the` m `leading monomials */`
   $\mathbf{N} \in \mathbb{K}[X]^{m \times m} \leftarrow \mathrm{LinNormalForm}(\mathbf{M}, \mathbf{F}, \mathbf{T}, <^{\mathbf{s}\text{-top}}, \beta, \mathbf{B})$
4. Return $\mathrm{diag}(X^{\delta_1}, \ldots, X^{\delta_m}) - \mathbf{N}$

4.3.3 Computing reduced Gröbner relation bases

Now, we give the general algorithm for the multivariate case, with $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X_1, \ldots, X_r]$. Let $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$ be pairwise commuting matrices in $\mathbb{K}^{D\times D}$, let $\mathbf{F} \in \mathbb{K}^{m\times D}$, and let $\prec$ be a monomial order on $\mathbb{K}[\boldsymbol{X}]^{1\times m}$. In what follows, we use some tools introduced in Section 1.3.5, concerning the border of the $\prec$-monomial basis of $\mathbb{K}[\boldsymbol{X}]^{1\times m}/\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$ and the minimal generators of the monomial submodule $\operatorname{in}_{\prec}(\operatorname{Syz}_{\mathbf{M}}(\mathbf{F}))$.

Our approach to compute the $\prec$-reduced Gröbner relation basis $\mathcal{G}$ of $\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$ is the same as in the univariate case. Namely, since Algorithms 1 and 2 provide us with efficient methods to compute the $\prec$-monomial basis $\mathcal{E}$ and $\prec$-normal forms modulo $\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$, we use $\mathcal{E}$ to find the set $\mathcal{L}$ formed by the $\prec$-leading terms of the polynomials in $\mathcal{G}$, and then we deduce $\mathcal{G}$ as $\mathcal{G} = \{f - \operatorname{nf}_{\prec}(f), f \in \mathcal{L}\}$.

In the algorithm below, the computation of

$$\mathcal{L} = \{\operatorname{in}_{\prec}(\mathbf{p}), \mathbf{p} \in \mathcal{G}\} = \{\boldsymbol{X}^{\boldsymbol{\delta}_j}, 1 \leqslant j \leqslant s\}$$

can be done from $\mathcal{E} = \{\varepsilon_1, \ldots, \varepsilon_\Delta\}$ as explained in Section 1.3.5. In short, we first compute the set of multiples $\mathcal{S} = \{X_k\varepsilon_j, 1 \leqslant k \leqslant r, 1 \leqslant j \leqslant \Delta\} \cup \{\mathbf{c}_i, 1 \leqslant i \leqslant m \mid \mathbf{c}_i \notin \mathcal{E}\}$, from which we then deduce the border $\mathcal{B} = \mathcal{S} - \mathcal{E}$. The latter is a set of generating monomials for the monomial submodule $\operatorname{in}_{\prec}(\operatorname{Syz}_{\mathbf{M}}(\mathbf{F}))$. Since $\mathcal{L}$ is the set of minimal generators of $\operatorname{in}_{\prec}(\operatorname{Syz}_{\mathbf{M}}(\mathbf{F}))$, it can be found from $\mathcal{B}$ by removing all monomials in $\mathcal{B}$ which are divisible by another monomial in $\mathcal{B}$. We recall that a monomial $\boldsymbol{X}^{\mathbf{e}}\mathbf{c}_i$ divides another monomial $\boldsymbol{X}^{\mathbf{e}'}\mathbf{c}_{i'}$ if $i = i'$ and $\boldsymbol{X}^{\mathbf{e}}$ divides $\boldsymbol{X}^{\mathbf{e}'}$.

Unlike in the univariate case, in this context the number of generators s is not known in advance; while it is at least m, it may also be much larger. We have the bound $s \leqslant r\Delta + m$, which follows for example from the computation of $\mathcal{L}$ above since $\mathcal{L} \subseteq \mathcal{S}$ with $\operatorname{Card}(\mathcal{S}) \leqslant r\Delta + m$; in the cost analysis below, we will use a similar argument. The s normal forms of the monomials in $\mathcal{L}$ can be computed efficiently using Algorithm 2, up to the preliminary computation of the submatrix $\mathbf{T}$ of $\mathcal{K}_{\mathbf{s},\beta}(\mathbf{M}, \mathbf{F})$ formed by the rows with indices in $\{\phi_{\prec,\boldsymbol{\beta}}(f), f \in \mathcal{L}\}$.

Proposition 4.19. *Algorithm 4 is correct and uses*

$$\begin{aligned}&\mathcal{O}\big(mD^{\omega-1} + D^{\omega}\log\big(2^r\textstyle\prod_{1\leqslant k\leqslant r}\min(\beta_k, \Delta)\big)\big)\\ &\quad\subseteq \mathcal{O}(mD^{\omega-1} + rD^{\omega}\log(\Delta))\end{aligned}$$

operations in $\mathbb{K}$.

Proof. Concerning correctness, the construction of $\mathbf{B}$ ensures that after Step **2.d**, the rows of $\mathbf{T}_k$ are the rows $\phi_{\prec,\boldsymbol{\beta}}(X_k^{-1}\boldsymbol{X}^{\mathbf{e}_{k,j}}\mathbf{c}_{i_{k,j}})$ of $\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M}, \mathbf{F})$. Therefore, by definition of the latter matrix, after Step **2.e** the rows of $\mathbf{T}_k$ are the rows $\mathbf{F}_{i_{k,j},*}\mathbf{M}^{\mathbf{e}_{k,j}}$. Then, Proposition 4.16 implies that the row j of $\mathbf{N}_k$ computed at Step **3** is the normal form $\operatorname{nf}_{\prec}(\boldsymbol{X}^{\mathbf{e}_{k,j}}\mathbf{c}_{i_{k,j}})$ with respect to the module $\operatorname{Syz}_{\mathbf{M}}(\mathbf{F})$. This shows the correctness of the algorithm since, as explained above, the $\prec$-reduced Gröbner relation basis is $\{f - \operatorname{nf}_{\prec}(f), f \in \mathcal{L}\}$.

Concerning the cost bound, Step **1** uses $\mathcal{O}\big(mD^{\omega-1} + D^{\omega}\log\big(\prod_{1\leqslant k\leqslant r}\min(\beta_k, \Delta)\big)\big)$ operations, according to Proposition 4.13. Then, at the iteration k of the For loop, the multiplication at Step **2.e** involves the $s_k \times D$ matrix $\mathbf{T}_k$ and the $D \times D$ matrix $\mathbf{M}_k$.

Algorithm 4 – LinRelBas
(Reduced Gröbner relation bases via linear algebra)

Input:

- pairwise commuting matrices $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$ in $\mathbb{K}^{D\times D}$,
- matrix $\mathbf{F} \in \mathbb{K}^{m\times D}$,
- monomial order $\prec$ on $\mathbb{K}[X_1, \ldots, X_r]^{1\times m}$,
- degree bounds $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_r) \in \mathbb{Z}^r_{>0}$ such that $\mathcal{H}_{\boldsymbol{\beta}}$ (defaults to $\boldsymbol{\beta} = (D+1, \ldots, D+1)$).

Output: the $\prec$-reduced Gröbner relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

1. `/* Compute monomial basis and multi-Krylov submatrix */`
   $\phi_{\prec,\boldsymbol{\beta}} \leftarrow$ indexing function as in Definition 4.2
   $\hat{\boldsymbol{\beta}} \leftarrow (2^{\lceil \log_2(\beta_1)\rceil}, \ldots, 2^{\lceil \log_2(\beta_r)\rceil})$
   $(\rho_1, \ldots, \rho_\Delta), \mathbf{B} \leftarrow$ KrylovRankProf$(\mathbf{M}, \mathbf{F}, \prec, \hat{\boldsymbol{\beta}})$
   $\mathcal{E} \leftarrow$ the $\prec$-monomial basis $\{\phi^{-1}_{\prec,\boldsymbol{\beta}}(\rho_k), 1 \leqslant k \leqslant \Delta\}$
2. `/* Compute leading monomials and their linearizations */`
   $\mathcal{L} \leftarrow$ minimal generating set for $\mathrm{in}_\prec(\mathrm{Syz}_{\mathbf{M}}(\mathbf{F}))$ deduced from $\mathcal{E}$
   $\mathcal{L}_0 \leftarrow \mathcal{L} \cap \{\mathbf{c}_i, 1 \leqslant i \leqslant m\}$
   write $\mathcal{L}_0 = \{\mathbf{c}_{i_{0,j}}, 1 \leqslant j \leqslant s_0\}$
   For k from 1 to r
   a. $\mathcal{L}_k \leftarrow \{f \in \mathcal{L} - (\mathcal{L}_0 \cup \cdots \cup \mathcal{L}_{k-1}) \mid X_k \text{ divides } f \text{ and } X_k^{-1} f \in \mathcal{E}\}$
   b. write $\mathcal{L}_k = \{\boldsymbol{X}^{\mathbf{e}_{k,j}} \mathbf{c}_{i_{k,j}}, 1 \leqslant j \leqslant s_k\}$
   c. For j from 1 to s_k: $\mu_j \leftarrow$ index such that $\rho_{\mu_j} = \phi_{\prec,\boldsymbol{\beta}}(X_k^{-1} \boldsymbol{X}^{\mathbf{e}_{k,j}} \mathbf{c}_{i_{k,j}})$
   d. $\mathbf{T}_k \leftarrow$ matrix formed by the rows $\mu_1, \ldots, \mu_{s_k}$ of $\mathbf{B}$, in this order
   e. $\mathbf{T}_k \leftarrow \mathbf{T}_k \mathbf{M}_k$

   $\mathbf{T} \leftarrow [\mathbf{T}_0^\mathsf{T} \mid \mathbf{T}_1^\mathsf{T} \mid \cdots \mid \mathbf{T}_r^\mathsf{T}]^\mathsf{T}$
3. `/* Compute normal forms of the` s `leading monomials */`
   $\mathbf{N} \in \mathbb{K}[X_1, \ldots, X_r]^{s\times m} \leftarrow$ LinNormalForm$(\mathbf{M}, \mathbf{F}, \mathbf{T}, \prec, \boldsymbol{\beta}, \mathbf{B})$
   write $\mathbf{N} = [\mathbf{N}_0^\mathsf{T} \mid \mathbf{N}_1^\mathsf{T} \mid \cdots \mid \mathbf{N}_r^\mathsf{T}]^\mathsf{T}$ with $\mathbf{N}_k \in \mathbb{K}[X_1, \ldots, X_r]^{s_k\times m}$
4. Denote by $\mathbf{N}_{k,j,*}$ the row j of $\mathbf{N}_k$ `// normal form of` $\boldsymbol{X}^{\mathbf{e}_{k,j}} \mathbf{c}_{i_{k,j}}$
   Return $\bigcup_{0\leqslant k\leqslant r} \{\boldsymbol{X}^{\mathbf{e}_{k,j}} \mathbf{c}_{i_{k,j}} - \mathbf{N}_{k,j,*}, 1 \leqslant j \leqslant s_k\}$

Since we have by definition $s_k = \mathrm{Card}(\mathcal{L}_k) \leqslant \mathrm{Card}(\mathcal{E}) = \Delta \leqslant D$, this multiplication is performed in $\mathcal{O}(D^\omega)$ operations; over the r iterations, this leads to a total of $\mathcal{O}(rD^\omega) = \mathcal{O}(D^\omega \log(2^r))$ operations. Finally, the cost for computing normal forms at Step **3** is in $\mathcal{O}(\Delta^{\omega-1}(D+s)) \subseteq \mathcal{O}(mD^{\omega-1} + D^\omega)$ according to Proposition 4.16. ☙

5

Computing multiplication matrices from a Gröbner basis

In this chapter, we give the details of a fast algorithm to compute multiplication matrices from a Gröbner basis (Problem 6). For this, we make an assumption on the input ideal, which is detailed in Section 5.1. We stated our result and gave a comparison with existing algorithms in Section 2.2.3.

5.1 Structural properties of the monomial basis

In this section, we consider the multivariate polynomial ring $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X_1, \ldots, X_r]$, and a zero-dimensional ideal $\mathcal{I}$ of $\mathbb{K}[\boldsymbol{X}]$ of degree D. Using the terminology in Section 1.3.5, this means that $\mathcal{I}$ has finite codimension D as a $\mathbb{K}$-vector subspace of $\mathbb{K}[\boldsymbol{X}]$.

For a given monomial order $\prec$ on $\mathbb{K}[\boldsymbol{X}]$, our goal in this section is to present basic properties about the shape of the $\prec$-monomial basis of $\mathbb{K}[\boldsymbol{X}]/\mathcal{I}$, and their consequences on the computation of the multiplication matrices from a Gröbner basis. To obtain a fast algorithm, we will use additional properties which derive from the assumption that the generic initial ideal of $\mathcal{I}$ is Borel-fixed; this is stated formally at the end of this section.

Let us denote by $\mathcal{E} = \{\varepsilon_1, \ldots, \varepsilon_D\}$ the $\prec$-monomial basis of $\mathbb{K}[\boldsymbol{X}]/\mathcal{I}$ (see Section 1.3.2). As in Section 1.3.5, we consider the set of monomials that are obtained from those in $\mathcal{E}$ by multiplication by one of the variables:

$$\mathcal{S} = \{X_k \varepsilon_j, 1 \leqslant k \leqslant r, 1 \leqslant j \leqslant D\}.$$

Then, if $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_r)$ are the sought multiplication matrices, for $1 \leqslant k \leqslant r$ the row j of $\mathbf{M}_k$ is given by the coefficients of the $\prec$-normal form $\mathrm{nf}_{\prec}(X_k \varepsilon_j)$ in the basis $\mathcal{E}$.

Assume that we are given the $\prec$-reduced Gröbner basis $\mathcal{G}$ of $\mathcal{I}$, which is a convenient tool to compute such $\prec$-normal forms and thus the matrices $\mathbf{M}$. Among the $\prec$-normal forms of the monomials in $\mathcal{S}$, some are easier to handle than others.

More precisely, the monomials in $\mathcal{S}$ can be divided into three disjoint categories:

$$\mathcal{S} = (\mathcal{S} - \mathcal{B}) \cup \mathcal{L} \cup (\mathcal{B} - \mathcal{L}),$$

where $\mathcal{B} = \mathcal{S} - \mathcal{E}$ is the border (see Section 1.3.5) and $\mathcal{L} = \{\mathrm{in}_{\prec}(\mathbf{g}), \mathbf{g} \in \mathcal{G}\} \subseteq \mathcal{B}$ are the minimal generators of $\mathrm{in}_{\prec}(\mathcal{I})$.

We first remark that $\mathcal{S} - \mathcal{B}$ is included in $\mathcal{E}$; in fact, we have $\mathcal{E} = \{1\} \cup (\mathcal{S} - \mathcal{B})$. As a result, the monomials in $\mathcal{S} - \mathcal{B}$ are their own $\prec$-normal forms, and the corresponding rows of the multiplication matrices are coordinate rows that are obtained for free.

Besides, the monomials in the second set $\mathcal{L}$ are the $\prec$-initial terms of the generators in $\mathcal{G}$. As such, we have $\mathcal{G} = \{f - \mathrm{nf}_{\prec}(f), f \in \mathcal{L}\}$ and thus the $\prec$-normal forms of the monomials in $\mathcal{L}$ can be computed from $\mathcal{G}$ using at most $D\mathrm{Card}(\mathcal{S}) \leqslant rD^2$ computations of opposites in $\mathbb{K}$. By opposite we mean having on input $\alpha \in \mathbb{K}$ and computing $-\alpha$.

We conclude that, to obtain the multiplication matrices, the main computational work is the computation of the $\prec$-normal forms of the monomials in $\mathcal{B} - \mathcal{L}$.

To the best of our knowledge, it is currently unknown how to achieve efficient computation of these normal forms with a sub-subic complexity, unless some assumptions are made to obtain additional properties about the structure of the monomial basis.

In [FGHR14], it is showed how the Moreno-Socias conjecture for generic ideals [MS91, MS03a] yields such a property [FGHR14, Proposition 7], leading to the efficient computation of the multiplication matrix of the smallest variable X_r. We remark that [FGHR14] focuses on the case of *Shape Position* ideals, and in this context one does not have to compute the other multiplication matrices.

In this reference, the non-generic case is also dealt with [FGHR14, Section 4.2], using a generalization of [FGHR14, Proposition 7] which is known as a property of *Borel-fixedness* of the initial ideal (see Lemma 5.1). This property holds after a random linear change of variables; indeed, a theorem of Galligo and Bayer-Stillman states that, if the order $\prec$ refines the degree, then the generic $\prec$-initial ideal of $\mathcal{I}$ is Borel-fixed (see for example [Eis95, Section 15.9] for more details). We note that the latter result is not conjectural.

Following this line of work, we will make the assumption that the $\prec$-initial ideal of $\mathcal{I}$ is Borel-fixed. We refer to [MS05, Chapter 2] for a definition; here, we are mainly interested in the characterization stated in Lemma 5.1 below. It gives a structural property which is the key behind the efficiency of our algorithms.

Lemma 5.1 ([MS05, Proposition 2.3])**.** *Assume that the characteristic of $\mathbb{K}$ is zero. Then, a monomial ideal $\mathcal{J} \subset \mathbb{K}[X]$ is Borel-fixed if and only if for any $f \in \mathcal{J}$ that is divisible by X_j then $\frac{X_i}{X_j} f \in \mathcal{J}$ for all $i < j$.*

For an illustration, we note that the monomial ideal in Fig. 1.2 has this property. The following sections extend the work in [FGHR14] concerning the computation of $\mathbf{M}_r$, by exploiting the Borel-fixed property to further compute all other multiplication matrices efficiently.

5.2 The case of two variables

In this section, we focus on the case of two variables, with $\mathcal{I}$ being an ideal of $\mathbb{K}[X, Y]$.

In what follows, we use notation from the previous section. Furthermore, for any set of monomials $\mathcal{A} \subset \mathbb{K}[X, Y]$, we write $\mathrm{nf}_{\prec}(\mathcal{A})$ to denote the set of $\prec$-normal forms of the

monomials in $\mathcal{A}$. Then, our goal is to compute $\mathrm{nf}_{\prec}(\mathcal{B} - \mathcal{L})$, where

$$\mathcal{B} = (\{X\varepsilon_j, 1 \leqslant j \leqslant D\} \cup \{Y\varepsilon_j, 1 \leqslant j \leqslant D\}) - \{\varepsilon_j, 1 \leqslant j \leqslant D\}.$$

In our algorithm, we will first compute the multiplication matrix $\mathbf{M}_Y$, and then $\mathbf{M}_X$. We have the following result.

Lemma 5.2. *Let $\prec$ be a monomial order on $\mathbb{K}[X, Y]$, and let $\mathcal{G}$ be a $\prec$-reduced Gröbner basis defining a zero-dimensional ideal $\mathcal{I} \in \mathbb{K}[X, Y]$ of degree D. Then,*

(i) knowing $\mathbf{M}_Y$, one can compute $\mathbf{M}_X$ using $\mathcal{O}(D^\omega \log(D))$ operations in $\mathbb{K}$;

(ii) assuming $\mathbb{K}$ has characteristic zero and that $\mathrm{in}_{\prec}(\mathcal{I})$ is Borel-fixed, then

$$\{Y\varepsilon_j, 1 \leqslant j \leqslant D\} \subset \mathcal{E} \cup \mathcal{L},$$

and $\mathbf{M}_Y$ can be obtained via the computation of $\mathcal{O}(D^2)$ opposites of field elements.

Proof. (i) According to Section 5.1, the rows of $\mathbf{M}_X$ corresponding to normal forms in $\mathrm{nf}_{\prec}(\mathcal{E} \cup \mathcal{L})$ are obtained in $\mathcal{O}(D^2)$ operations in $\mathbb{K}$. It remains to compute its rows that are given by $\mathrm{nf}_{\prec}(\mathcal{B}_X)$, where

$$\mathcal{B}_X = \{X\varepsilon_j, 1 \leqslant j \leqslant D\} - \{\varepsilon_j, 1 \leqslant j \leqslant D\};$$

for this, we give details on the structure of $\mathcal{B}_X$.

We write $\mathcal{L} = \{X^{\alpha_j} Y^{\beta_j}, 1 \leqslant j \leqslant s\}$ with $(\alpha_{j+1}, \beta_{j+1}) \prec_{\mathrm{lex}} (\alpha_j, \beta_j)$ for $1 \leqslant j \leqslant s$; notice that (α_j) is strictly decreasing with $\alpha_s = 0$, and (β_j) is strictly increasing with $\beta_1 = 0$. Then, we have

$$\mathcal{B}_X = \{X^{\alpha_j} Y^{\beta_j + k}, 1 \leqslant k < \beta_{j+1} - \beta_j, 1 \leqslant j \leqslant s - 1\}.$$

Now let $\mathbf{v}_j \in \mathbb{K}^{1 \times D}$ be the vector which gives the coefficients of $\mathrm{nf}_{\prec}(X^{\alpha_j} Y^{\beta_j})$ in the basis $\mathcal{E}$, for $1 \leqslant j \leqslant s$. Since $\{\mathbf{v}_1, \ldots, \mathbf{v}_s\}$ corresponds to $\mathrm{nf}_{\prec}(\mathcal{L})$, these vectors are among the first set of rows of $\mathbf{M}_X$ that have already been computed. Then, the normal forms $\mathrm{nf}_{\prec}(\mathcal{B}_X)$ can be obtain from the fact that

$$\mathrm{nf}_{\prec}(X^{\alpha_j} Y^{\beta_j + k}) \text{ is given by } \mathbf{v}_j \mathbf{M}_Y^k \text{ for } 1 \leqslant k < \beta_{j+1} - \beta_j, 1 \leqslant j \leqslant s - 1.$$

We will study this kind of computation separately below in Section 5.3. In short, the cost bound in Lemma 5.3 and the inequality $\sum_{1 \leqslant j \leqslant s-1} \beta_{j+1} - \beta_j = \beta_s \leqslant D$ ensure that $\mathrm{nf}_{\prec}(\mathcal{B}_X)$ can be computed using $\mathcal{O}(D^\omega \log(D))$ operations in $\mathbb{K}$.

(ii) This item is proved in [FGHR13, Section 7], although under slightly different assumptions; we still give a proof here for completeness.

Assume that for some j we have $Y\varepsilon_j \notin \mathcal{E} \cup \mathcal{L}$. First, $Y\varepsilon_j \notin \mathcal{E}$ implies that $Y\varepsilon_j \in \mathrm{in}_{\prec}(\mathcal{I})$. Then, $Y\varepsilon_j$ is a multiple of some $f \in \mathcal{L}$: there are exponents $\alpha, \beta \geqslant 0$ such that $Y\varepsilon_j = fX^\alpha Y^\beta$. The fact that $Y\varepsilon_j \notin \mathcal{L}$ implies that either $\alpha > 0$ or $\beta > 0$.

If $\beta > 0$ then we would have $\varepsilon_j = fX^\alpha Y^{\beta-1} \in \mathrm{in}_{\prec}(\mathcal{I})$, which is not the case since $\varepsilon_j \in \mathcal{E}$. Hence $Y\varepsilon_j = fX^\alpha$ with $\alpha > 0$. Then, $fX^{\alpha-1} \in \mathrm{in}_{\prec}(\mathcal{I})$, and since by assumption $\mathbb{K}$ is of characteristic zero and $\mathrm{in}_{\prec}(\mathcal{I})$ is Borel-fixed, Lemma 5.1 implies that $\frac{X}{Y} fX^{\alpha-1} = \frac{f}{Y} X^\alpha = \varepsilon_j \in \mathrm{in}_{\prec}(\mathcal{I})$. Again, this is a contradiction since $\varepsilon_j \in \mathcal{E}$. Therefore there is no such j, and we have the announced inclusion. ∎

As a consequence, if $\mathbb{K}$ has characteristic zero and $\mathrm{in}_{\prec}(\mathcal{I})$ is Borel-fixed, then the multiplication matrices $\mathbf{M}_X$ and $\mathbf{M}_Y$ can be computed from the $\prec$-reduced Gröbner basis $\mathcal{G}$ using $\mathcal{O}(D^\omega \log(D))$ operations in $\mathbb{K}$.

5.3 Computing rows of a Krylov matrix

In this section, we detail a simple method for the computation of a collection of vector-matrix products of the form $\mathbf{v}\mathbf{M}^e$ (Algorithm 5), obtaining efficiency via the use of repeated squaring of the matrix $\mathbf{M}$.

In this algorithm we use the following notation: for a matrix $\mathbf{N} \in \mathbb{K}^{m\times D}$ whose rows are indexed by a set $\mathcal{P}$ of cardinality m, then for any subset $\mathcal{P}' \subset \mathcal{P}$, the notation $\mathrm{Submatrix}(\mathbf{N}, \mathcal{P}')$ stands for the submatrix of $\mathbf{N}$ with the rows whose indices are in $\mathcal{P}'$. Furthermore, when we write an assignment operation $\mathrm{Submatrix}(\mathbf{N}, \mathcal{P}') \leftarrow \mathbf{A}$ for some matrix $\mathbf{A} \in \mathbb{K}^{\mathrm{Card}(\mathcal{P}')\times D}$, this does modify the entries of $\mathbf{N}$.

Lemma 5.3. *Let $\mathbf{M} \in \mathbb{K}^{D\times D}$, let $\mathbf{v}_1, \dots, \mathbf{v}_n \in \mathbb{K}^{1\times D}$, and let $\gamma_1, \dots, \gamma_n \in \mathbb{Z}_{>0}$, for some $n \in \mathbb{Z}_{>0}$. Then, Algorithm 5 computes the row vectors $\cup_{1\leqslant j\leqslant n}\{\mathbf{v}_j\mathbf{M}^e,\ 1 \leqslant e \leqslant \gamma_j\}$ using*

$$\mathcal{O}\big(D^\omega \log(1+\mu) + D^{\omega-1} d \log(1+\mu n/d)\big)$$

operations in $\mathbb{K}$, where $d = \gamma_1 + \cdots + \gamma_n$ and $\mu = \max_{1\leqslant i\leqslant n} \gamma_i$.

Proof. Here, we use notation from the algorithm, and we also denote by $\delta = d/n$ the average of $\gamma_1, \dots, \gamma_n$. The product at Step **5** uses $\mathcal{O}(D^{\omega-1}(D+n)) \subseteq \mathcal{O}(D^\omega + D^{\omega-1}n)$ field operations. Over all iterations of the While loop, Step **6.a** $\mathcal{O}(D^\omega \log(\mu))$ operations. Concerning Step **6.c**, we have the following bounds:

- For $1 \leqslant i \leqslant \log_2(\delta)$, we are multiplying an $\mathrm{Card}(\overline{\mathcal{P}}_{i-1}) \times D$ matrix and a $D \times D$ matrix, with $\mathrm{Card}(\overline{\mathcal{P}}_{i-1}) \leqslant 2^{i-1}n$. This costs $\mathcal{O}(D^{\omega-1}(D + 2^i n))$ for one i, and thus $\mathcal{O}(D^{\omega-1}(D\log(\delta) + d))$ overall.
- For iterations with $\log(\delta) < i \leqslant \lceil \log(\mu) \rceil$, we have a similar multiplication with the left-hand side having $\mathrm{Card}(\overline{\mathcal{P}}_{i-1}) \leqslant \mathrm{Card}(\mathcal{P}) = d$ rows. Overall, this uses $\mathcal{O}(D^{\omega-1}(D+d)\log(\mu/\delta))$ field operations.

Summing these bounds leads to the announced cost, noting that the ones in the logarithms are added to take into account the cost of Step **5** when $\gamma_1 = \cdots = \gamma_n = 1$. ∎

5.4 Computing the multiplication matrices

In this section, we generalize our results of Section 5.2 to the case of r variables. We follow a similar strategy, computing iteratively the matrices $\mathbf{M}_r, \mathbf{M}_{r-1}, \dots, \mathbf{M}_1$.

Assuming that $\mathrm{in}_{\prec}(\mathcal{I})$ is Borel-fixed and using the same ideas as in Section 5.2, we can indeed obtain $\mathbf{M}_r$ by computing the normal forms of the monomials in $\mathcal{E} \cup \mathcal{L}$, and we can also obtain $\mathbf{M}_{r-1}$ by iterating $\mathbf{M}_r$ on these normal forms.

Algorithm 5 – KrylovEval
(Computing rows of a Krylov matrix)

Input:

- a matrix $\mathbf{M} \in \mathbb{K}^{D \times D}$ for some $D \in \mathbb{Z}_{>0}$,
- row vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n \in \mathbb{K}^{1 \times D}$ for some $n \in \mathbb{Z}_{>0}$,
- bounds $\gamma_1, \ldots, \gamma_n \in \mathbb{Z}_{>0}$.

Output: the matrix $\mathbf{N} \in \mathbb{K}^{(\gamma_1 + \cdots + \gamma_n) \times D}$ whose row $\gamma_1 + \cdots + \gamma_{j-1} + e$ is equal to $\mathbf{v}_j \mathbf{M}^e$, for $1 \leqslant e \leqslant \gamma_j$, and for $1 \leqslant j \leqslant n$.

1. $\mathcal{P} \leftarrow \{(e, j), 1 \leqslant e \leqslant \gamma_j, 1 \leqslant j \leqslant n\}$
2. $\mathbf{N} \leftarrow \mathbf{0} \in \mathbb{K}^{(\gamma_1 + \cdots + \gamma_n) \times D}$ with its rows being indexed by $\mathcal{P}$ sorted in lexicographic order
3. $\mathbf{M}^{(0)} \leftarrow \mathbf{M}$
4. $\mathcal{P}_0 \leftarrow \{(1, j), 1 \leqslant j \leqslant n\}$
5. $\mathrm{Submatrix}(\mathbf{N}, \mathcal{P}_0) \leftarrow \begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_n \end{bmatrix} \mathbf{M}^{(0)}$
6. For i from 1 to $\lceil \log_2(\max_i \gamma_i) \rceil$:
   - **a.** If $i = 1$ then $\mathbf{M}^{(1)} \leftarrow \mathbf{M}$ else $\mathbf{M}^{(i)} \leftarrow \mathbf{M}^{(i-1)} \cdot \mathbf{M}^{(i-1)}$
   - **b.** $\mathcal{P}_i \leftarrow \{(e, j), 2^{i-1} < e \leqslant 2^i, 1 \leqslant j \leqslant n\}$
   - **c.** $\overline{\mathcal{P}}_{i-1} = \mathcal{P} \cap (\mathcal{P}_0 \cup \cdots \cup \mathcal{P}_{i-1}) = \mathcal{P} \cap \{(e, j), 1 \leqslant e \leqslant 2^{i-1}, 1 \leqslant j \leqslant n\}$
   - **d.** $\hat{\mathcal{P}}_{i-1} = \{(e + 2^{i-1}, j), (e, j) \in \overline{\mathcal{P}}_{i-1}\}$ // $(\mathcal{P} \cap \mathcal{P}_i) \subset \hat{\mathcal{P}}_{i-1} \subset \mathcal{P}_i$
   - **e.** $\mathbf{N}^{(i)} \leftarrow \mathrm{Submatrix}(\mathbf{N}, \overline{\mathcal{P}}_{i-1})\, \mathbf{M}^{(i)}$ with the rows of $\mathbf{N}^{(i)}$ being indexed by $\hat{\mathcal{P}}^{(i-1)}$ sorted in lexicographic order
   - **f.** $\mathrm{Submatrix}(\mathbf{N}, \mathcal{P} \cap \mathcal{P}_i) \leftarrow \mathrm{Submatrix}(\mathbf{N}^{(i)}, \mathcal{P} \cap \mathcal{P}_i)$
7. Return $\mathbf{N}$

However, $\mathbf{M}_{r-2}$ cannot be obtained by simply iterating $\mathbf{M}_{r-1}$ on the normal forms in $\mathrm{nf}_{\prec}(\mathcal{E} \cup \mathcal{L})$ and those given by the rows of $\mathbf{M}_{r-1}$. The reason is that some of the normal forms which constitute the rows of $\mathbf{M}_{r-2}$ are actually obtained by iterating $\mathbf{M}_r$ on the normal forms in $\mathrm{nf}_{\prec}(\mathcal{L})$.

Thus we will change our focus, from the computation of the multiplication matrices, to that of the normal forms which we can obtain from the known multiplication matrices. Roughly, our algorithm is as follows. We start from $\mathcal{S}_r = \mathcal{E} \cup \mathcal{L}$, for which we have seen how to efficiently compute $\mathrm{nf}_{\prec}(\mathcal{S}_r)$. The Borel-fixed property ensures that these normal forms contains those giving $\mathbf{M}_r$. Then, we consider the monomials $\mathcal{S}_{r-1}$ that can be obtained by iterating X_r on $\mathcal{E} \cup \mathcal{L}$, and using $\mathbf{M}_r$ we compute their normal forms $\mathrm{nf}_{\prec}(\mathcal{S}_{r-1})$. Thanks to the Borel-fixed property again, this gives in particular $\mathbf{M}_{r-1}$, but also additional normal forms which give some rows of multiplication matrices $\mathbf{M}_i$ for $i < r-1$. Then, we continue this process until $i = 1$: at this stage, we have covered the whole set of monomials $\mathcal{S}$ and we thus have all the normal forms in $\mathrm{nf}_{\prec}(\mathcal{S})$. We recall that $\mathrm{nf}_{\prec}(\mathcal{S})$ precisely corresponds to the rows of the multiplication matrices $\mathbf{M}_1, \ldots, \mathbf{M}_r$.

More precisely, we consider the following subsets of $\mathcal{S} \cup \{1\} = \mathcal{E} \cup \mathcal{B}$, of which we will compute the normal forms iteratively:

- for $i = r$, the set of monomials
$$\hat{\mathcal{S}}_r = \mathcal{E} \cup \mathcal{L} \ \subset \mathcal{E} \cup \mathcal{B}\,,$$
whose normal forms are trivial or directly obtained from the $\prec$-reduced Gröbner basis $\mathcal{G}$;
- for $1 \leqslant i < r$, the set of monomials in $\mathcal{E} \cup \mathcal{B}$ which are obtained from $\mathcal{E} \cup \mathcal{L}$ through multiplication by a monomial in the variables $X_{i+1}, \ldots, X_r$:
$$\hat{\mathcal{S}}_i = \{X_{i+1}^{e_{i+1}} \cdots X_r^{e_r} f, e_{i+1}, \ldots, e_r \geqslant 0, f \in \mathcal{E} \cup \mathcal{L}\} \cap (\mathcal{E} \cup \mathcal{B}),$$
whose normal forms can thus be obtained from those in $\mathrm{nf}_{\prec}(\mathcal{E} \cup \mathcal{L})$ through multiplication by powers of $\mathbf{M}_{i+1}, \ldots, \mathbf{M}_r$, in the case where these matrices are known.

Then, we define $\mathcal{S}_r = \hat{\mathcal{S}}_r = \mathcal{E} \cup \mathcal{L}$ for $i = r$, and $\mathcal{S}_i = \hat{\mathcal{S}}_i - \hat{\mathcal{S}}_{i+1}$ for $1 \leqslant i < r$. Therefore $\hat{\mathcal{S}}_i$ is the disjoint union $\mathcal{S}_i \cup \cdots \cup \mathcal{S}_r$, and $\mathcal{S}_i$ is the set of monomials in $\mathcal{B} - \hat{\mathcal{S}}_{i+1}$ which can be obtained from $\mathcal{E} \cup \mathcal{L}$ through multiplication by a monomial in the variables $X_{i+1}, \ldots, X_r$ which does involve the variable X_{i+1}. That is, we can describe $\mathcal{S}_i$ as follows,

$$\begin{aligned}
\mathcal{S}_i &= \{X_{i+1}^{e_{i+1}} \cdots X_r^{e_r} f, e_{i+1} > 0, e_{i+2}, \ldots, e_r \geqslant 0, f \in \mathcal{E} \cup \mathcal{L}\} \cap (\mathcal{B} - \hat{\mathcal{S}}_{i+1}) \\
&= \{X_{i+1}^{e} f, e > 0, f \in \hat{\mathcal{S}}_{i+1}\} \cap (\mathcal{B} - \hat{\mathcal{S}}_{i+1}) \\
&= \{f \in \mathcal{B} - \hat{\mathcal{S}}_{i+1} \ \mid \ \exists e > 0, X_{i+1}^{-e} f \in \hat{\mathcal{S}}_{i+1}\}.
\end{aligned}$$

In particular, $\mathrm{nf}_{\prec}(\mathcal{S}_i)$ can be deduced from $\mathrm{nf}_{\prec}(\hat{\mathcal{S}}_{i+1})$ through multiplication by powers of $\mathbf{M}_{i+1}$, if the latter matrix is known.

We have

$$\begin{aligned}
\mathcal{B} - \mathcal{L} &= \mathcal{S}_1 \cup \cdots \cup \mathcal{S}_{r-1}, \\
\text{and } \ \mathcal{S} \cup \{1\} = \mathcal{E} \cup \mathcal{B} &= \mathcal{E} \cup \mathcal{L} \cup (\mathcal{B} - \mathcal{L}) = \mathcal{S}_1 \cup \cdots \cup \mathcal{S}_r,
\end{aligned}$$

so that $\mathrm{nf}_{\prec}(\mathcal{S})$ can be obtained by iteratively computing $\mathrm{nf}_{\prec}(\mathcal{S}_i)$ for i from r to 1. Our goal is to show that this strategy yields an efficient algorithm when $\mathrm{in}_{\prec}(\mathcal{I})$ is Borel-fixed.

Remark 5.4. One first question is whether the sets of monomials $(\mathcal{S}_i)_{1\leqslant i\leqslant r}$ can be efficiently determined. For example, this can be done while building $\mathcal{B}$ from $\mathcal{G}$, or afterwards by an iteration for i from r to 1 as in Algorithm 6. In any case, this does not involve field operations, but only integer comparisons, so that here the time for finding $(\mathcal{S}_i)_{1\leqslant i\leqslant r}$ is not taken into account in our cost bounds. ☕

Algorithm 6 – NextExpSet
(Computing the next exponent set $\mathcal{S}_i$)

Input:
- the border $\mathcal{B}$,
- the exponent set $\hat{\mathcal{S}}_{i+1} = \mathcal{S}_{i+1} \cup \cdots \cup \mathcal{S}_r$ for some $1 \leqslant i < r$.

Output: the exponent set $\mathcal{S}_i$.

1. Write $\hat{\mathcal{S}}_{i+1} = \{f_1, \ldots, f_N\}$
2. For j from 1 to N:
   a. $e \leftarrow 0$
   b. While $X_{i+1}^{e+1} f_j \in \mathcal{B} - \hat{\mathcal{S}}_{i+1}$:
      (i) $e \leftarrow e + 1$
   c. If $e > 0$:
      (i) $\mathcal{S}_i \leftarrow \mathcal{S}_i \cup \{X_{i+1} f_j, \ldots, X_{i+1}^{e} f_j\}$
3. Return $\mathcal{S}_i$

Now, we focus on the computation of the normal forms $\mathrm{nf}_{\prec}(\mathcal{S})$. The efficiency of our iterative algorithm is based on the following fact. As a consequence of $\mathrm{in}_{\prec}(\mathcal{I})$ being Borel-fixed, the structure of $\mathcal{S}$ is such that the knowledge of the normal forms $\mathrm{nf}_{\prec}(\mathcal{S}_i \cup \cdots \cup \mathcal{S}_r)$ directly gives the multiplication matrices $\mathbf{M}_i, \ldots, \mathbf{M}_r$. We first detail this claim in the following lemma.

Lemma 5.5. *Let $\prec$ be a monomial order on $\mathbb{K}[X_1, \ldots, X_r]$, and let $\mathcal{G}$ be a $\prec$-reduced Gröbner basis defining a zero-dimensional ideal $\mathcal{I} \in \mathbb{K}[X_1, \ldots, X_r]$ of degree D. Let us assume that $\mathbb{K}$ has characteristic zero and that $\mathrm{in}_{\prec}(\mathcal{I})$ is Borel-fixed. Then, we have*

$$\{X_i \varepsilon_j, 1 \leqslant j \leqslant D\} \quad \subset \quad \hat{\mathcal{S}}_i \quad \textit{for all } 1 \leqslant i \leqslant r;$$

in particular, the multiplication matrices $\mathbf{M}_i, \ldots, \mathbf{M}_r$ can be read off from $\mathrm{nf}_{\prec}(\hat{\mathcal{S}}_i)$.

Proof. This proof uses arguments similar to those in the proof of Lemma 5.2. For $i = r$, this inclusion is already proved in [FGHR13, Section 7]. For $i = 1$, the property is trivial since $\hat{\mathcal{S}}_1 = \mathcal{S} \cup \{1\} = \mathcal{E} \cup \mathcal{B}$.

Let $i \in \{1, \dots, r\}$. Then, for all $i' \in \{i+1, \dots, r\}$, the fact that $\{X_{i'}\varepsilon_j, 1 \leqslant j \leqslant D\} \subset \hat{\mathcal{S}}_{i'-1} \subset \hat{\mathcal{S}}_i$ implies that $\mathbf{M}_{i'}$ can be read off from $\mathrm{nf}_{\prec}(\hat{\mathcal{S}}_i)$. Now let us fix $j \in \{1, \dots, D\}$. As above we have $X_i\varepsilon_j \in \hat{\mathcal{S}}_{i-1}$, but our goal is to use the fact that $\mathrm{in}_{\prec}(\mathcal{I})$ is Borel-fixed, to prove that we actually have $X_i\varepsilon_j \in \hat{\mathcal{S}}_i$, or in other words, $X_i\varepsilon_j \notin \mathcal{S}_{i-1}$.

If $X_i\varepsilon_j \in \mathcal{E} \cup \mathcal{L} = \mathcal{S}_r$, then the conclusion follows from $\mathcal{S}_r \in \hat{\mathcal{S}}_i$. Let us now consider the case $X_i\varepsilon_j \notin \mathcal{E} \cup \mathcal{L}$. Thus $X_i\varepsilon_j \in \mathrm{in}_{\prec}(\mathcal{I})$, and therefore there exist exponents $(\alpha_k)_{1 \leqslant k \leqslant r}$ not all zero and a generating monomial $f \in \mathcal{L}$ such that $X_i\varepsilon_j = X_1^{\alpha_1} \cdots X_r^{\alpha_r} f$.

Suppose that there exists $k \in \{1, \dots, i\}$ such that $\alpha_k > 0$. If $k = i$ then from $\alpha_i > 0$ we have that ε_j is a multiple of f and thus $\varepsilon_j \in \mathrm{in}_{\prec}(\mathcal{I})$, which is not the case; hence $k < i$. If $i = 1$, there is no such k; otherwise, we have $X_i \frac{\varepsilon_j}{X_k} \in \mathrm{in}_{\prec}(\mathcal{I})$, and by Lemma 5.1 this also leads to $\frac{X_k}{X_i} X_i \frac{\varepsilon_j}{X_k} = \varepsilon_j \in \mathrm{in}_{\prec}(\mathcal{I})$. In any case, there is no such k, that is, we have $\alpha_1 = \cdots = \alpha_i = 0$. If $i = r$ this contradicts the definition of $(\alpha_k)_{1 \leqslant k \leqslant r}$ and therefore $X_i\varepsilon_j \in \mathcal{E} \cup \mathcal{L}$; otherwise this gives the conclusion $X_i\varepsilon_j = X_{i+1}^{\alpha_{i+1}} \cdots X_r^{\alpha_r} f \in \hat{\mathcal{S}}_i - (\mathcal{E} \cup \mathcal{L})$. ☙

Next we show how one can efficiently compute $\mathrm{nf}_{\prec}(\mathcal{S}_i)$ from $\mathrm{nf}_{\prec}(\hat{\mathcal{S}}_{i+1})$ when $\mathbf{M}_{i+1}$ is known. The cost bound we give for our approach depends on a parameter η_i, which is defined as follows. For $i \in \{1, \dots, r-1\}$, it is the largest exponent e such that there is a monomial $f \in \mathcal{B}$ satisfying $X_{i+1}^e f \in \mathcal{B}$:

$$\eta_i = \max\{e \in \mathbb{Z}_{\geqslant 0} \mid X_{i+1}^e f \in \mathcal{B} \text{ for some } f \in \mathcal{B}\}. \tag{5.1}$$

Note that we do not allow f to be in the monomial basis $\mathcal{E}$.

Lemma 5.6. *Let $\prec$ be a monomial order on $\mathbb{K}[X_1, \dots, X_r]$, and let $\mathcal{G}$ be a $\prec$-reduced Gröbner basis defining a zero-dimensional ideal $\mathcal{I} \in \mathbb{K}[X_1, \dots, X_r]$ of degree D. Given $i \in \{1, \dots, r-1\}$, $\hat{\mathcal{S}}_{i+1}$, and $\mathrm{nf}_{\prec}(\hat{\mathcal{S}}_{i+1})$, there is a deterministic algorithm which computes $\mathcal{S}_i$ and $\mathrm{nf}_{\prec}(\mathcal{S}_i)$ using $\mathcal{O}((D^\omega + D^{\omega-1}C_i)\log(\eta_i))$ operations in $\mathbb{K}$, where $C_i = \mathrm{Card}(\mathcal{S}_i)$.*

Proof. First, from Lemma 5.5 we know that the normal forms in $\mathrm{nf}_{\prec}(\hat{\mathcal{S}}_{i+1})$ contain all those that define the multiplication matrix $\mathbf{M}_{i+1}$, which we thus have without performing any additional field operation.

Then, $\mathcal{S}_i$ can be determined from $\hat{\mathcal{S}}_{i+1}$ without field operation as shown in Algorithm 6 (see also Remark 5.4). More precisely, we obtain $\mathcal{S}_i$ in the following form:

$$\mathcal{S}_i = \bigcup_{1 \leqslant j \leqslant N} \{X_{i+1}^e f_j, 1 \leqslant e \leqslant e_j\}, \tag{5.2}$$

where $e_1, \dots, e_N > 0$ and $\{f_j, 1 \leqslant j \leqslant N\} \subset \hat{\mathcal{S}}_{i+1}$. More precisely, $\{f_j, 1 \leqslant j \leqslant N\}$ is the subset of the monomials $f_j \in \hat{\mathcal{S}}_{i+1}$ that are such that $X_{i+1}^{e_j} f_j \in \mathcal{B} - \hat{\mathcal{S}}_{i+1}$. Remark that $e_1 + \cdots + e_N = \mathrm{Card}(\mathcal{S}_i) = C_i$.

Going to the normal forms, this directly gives

$$\mathrm{nf}_{\prec}(\mathcal{S}_i) = \bigcup_{1 \leqslant j \leqslant N} \{\mathbf{v}_j \mathbf{M}_{i+1}^e, 1 \leqslant e \leqslant e_j\}, \tag{5.3}$$

where $\mathbf{v}_j \in \mathbb{K}^{1 \times D}$ is the coefficient vector of f_j in the basis $\mathcal{E}$. Therefore, according to Lemma 5.3, $\mathrm{nf}_{\prec}(\mathcal{S}_i)$ can be computed using

$$\mathcal{O}((D^\omega + D^{\omega-1}(e_1 + \cdots + e_N))\log(\max_i e_i)) \subset \mathcal{O}((D^\omega + D^{\omega-1}C_i)\log(\eta_i))$$

operations in $\mathbb{K}$. ☛

Algorithm 7 – MulMat
(Multiplication matrices from reduced Gröbner basis)

Input:

- a monomial order $\prec$ on $\mathbb{K}[X_1, \ldots, X_r]$,
- a $\prec$-reduced Gröbner basis $\mathcal{G}$ defining a zero-dimensional ideal $\mathcal{I}$

Output: the multiplication matrices $\mathbf{M}_1, \ldots, \mathbf{M}_r$ in $\mathbb{K}[X_1, \ldots, X_r]/\mathcal{I}$.

1. Read $\mathcal{L}$ and $\mathcal{E} = \{\varepsilon_1, \ldots, \varepsilon_D\}$ from $\mathcal{G}$
2. $\mathcal{S} \leftarrow \{X_k \varepsilon_j, 1 \leqslant k \leqslant r, 1 \leqslant j \leqslant D\}$; $\mathcal{B} \leftarrow \mathcal{S} - \mathcal{E}$
3. $\hat{\mathcal{S}} \leftarrow \mathcal{E} \cup \mathcal{L}$; $\mathcal{N} \leftarrow \mathrm{nf}_{\prec}(\hat{\mathcal{S}})$
4. For i from $r-1$ to 1:
   - **a.** Read $\mathbf{M}_{i+1}$ from $\hat{\mathcal{S}}$
   - **b.** Compute $\mathcal{S}_i = \bigcup_{1 \leqslant j \leqslant N} \{X_{i+1}^e f_j, 1 \leqslant e \leqslant e_j\}$ as in Eq. (5.2), using Algorithm 6 on input $\mathcal{B}$ and $\hat{\mathcal{S}}$
   - **c.** Compute $\mathrm{nf}_{\prec}(\mathcal{S}_i) = \bigcup_{1 \leqslant j \leqslant N} \{\mathbf{v}_j \mathbf{M}_{i+1}^e, 1 \leqslant e \leqslant e_j\}$ as in Eq. (5.3), using Algorithm 5 on input $\mathbf{M}_{i+1}$, $(\mathbf{v}_1, \ldots, \mathbf{v}_N)$, and $(e_1, \ldots, e_N)$
   - **d.** $\hat{\mathcal{S}} \leftarrow \mathcal{S}_i \cup \hat{\mathcal{S}}$; $\mathcal{N} \leftarrow \mathrm{nf}_{\prec}(\mathcal{S}_i) \cup \mathcal{N}$
5. Read $\mathbf{M}_1$ from $\hat{\mathcal{S}}$
6. Return $\mathbf{M}_1, \ldots, \mathbf{M}_r$

The correctness of Algorithm 7 follows from the results and discussions in this section. The next proposition implies Theorem 2.14, and gives a more precise cost bound.

Proposition 5.7. *Let $\prec$ be a monomial order on $\mathbb{K}[X_1, \ldots, X_r]$, and let $\mathcal{G}$ be a $\prec$-reduced Gröbner basis defining a zero-dimensional ideal $\mathcal{I} \in \mathbb{K}[X_1, \ldots, X_r]$ of degree D. If $\mathbb{K}$ is of characteristic 0, and if $\mathrm{in}_{\prec}(\mathcal{I})$ is Borel-fixed, then Algorithm 7 solves Problem 6 using $\mathcal{O}(D^{\omega-1}\mathrm{Card}(\mathcal{S})\log(\eta)) \subseteq \mathcal{O}(rD^{\omega}\log(\eta))$ operations in $\mathbb{K}$, where*

$$\eta = \max\{e \in \mathbb{Z}_{\geqslant 0} \mid X_i^e f \in \mathcal{B} \text{ for some } f \in \mathcal{B} \text{ and some } 2 \leqslant i \leqslant r\}.$$

Proof. First, we compute $\hat{\mathcal{S}}_r = \mathcal{S}_r = \mathcal{E} \cup \mathcal{L}$ and $\mathrm{nf}_{\prec}(\hat{\mathcal{S}}_r)$ from $\mathcal{G}$, using $\mathcal{O}(rD^2)$ computations of opposites of field elements. Then, we iteratively apply Lemma 5.6 to obtain the remaining matrices. The overall cost bound is $\mathcal{O}(\sum_{1 \leqslant i \leqslant r-1}(D^{\omega} + D^{\omega-1}C_i)\log(\eta_i))$, with $C_i = \mathrm{Card}(\mathcal{S}_i)$. Using the upper bound $C_1 + \cdots + C_{r-1} \leqslant \mathrm{Card}(\mathcal{S}) \leqslant rD$ and the fact that $\max_i \eta_i = \eta$, we obtain that the claimed cost bound. ☛

Part III

Systems of linear modular univariate equations

Contents

Chapter 6 Preliminaries and ingredients **155**

6.1 Multiplication time functions for polynomials and polynomial matrices 155

6.2 Using the minimal degree to ensure uniform shift and output degrees . 156

6.3 Computing residuals for systems of linear modular equations 158

6.4 Iterative relation basis for a triangular multiplication matrix [BL00] . . 160

Chapter 7 Computing shifted Popov approximant bases **165**

7.1 Fast algorithms for almost uniform orders [GJV03] 166

7.2 Arbitrary orders: reduction to the case $n \in \mathcal{O}(m)$ 172

7.3 Fast approximant bases in Popov form with known minimal degree . . 175

7.4 Fast approximant bases in Popov form for arbitrary shifts 181

Chapter 8 Computing shifted Popov solution bases **185**

8.1 Fast algorithm via kernel bases when the minimal degree is known . . . 186

8.2 The case of one equation 190

8.2.1 Amplitude, splitting indices, and block triangular shape 192

8.2.2 Fast algorithm for a single equation 194

8.3 Fast solution bases in Popov form for arbitrary shifts 198

Chapter 9 Computing a solution via structured linear algebra **201**

9.1 Solving structured homogeneous linear systems 201

9.2 Reducing to solving a mosaic-Hankel linear system 204

9.3 Directly computing a solution via a Toeplitz-like system 208

Chapter 10 Coppersmith technique over the univariate polynomials **215**
10.1 The approach based on row reduction 215
10.2 Reducing to a system of linear modular equations 218
10.2.1 Introduction: the specific case $d = 1$ 218
10.2.2 The general case $d \geqslant 1$ 220

6

Preliminaries and ingredients

6.1 Multiplication time functions for polynomials and polynomial matrices

In what follows, we use the time function $d \mapsto \mathsf{M}(d)$ for polynomial multiplication over a field $\mathbb{K}$, as in [GG13, Section 8.3]: $\mathsf{M}(\cdot)$ is such that, for any positive real number d, two polynomials of degree at most d in $\mathbb{K}[X]$ can be multiplied using $\mathsf{M}(d)$ operations in $\mathbb{K}$.

Concerning assumptions on $\mathsf{M}(d)$, we will mainly restrict ourselves to the most widely used one, called the *superlinearity property*:

$$\mathcal{H}_{\mathsf{M}(\cdot)} \ : \quad \mathsf{M}(d) + \mathsf{M}(d') \leqslant \mathsf{M}(d + d') \ \text{ for any } d, d' > 0. \tag{6.1}$$

One may of course consider that $\mathsf{M}(\cdot)$ is at most quadratic, that is, $\mathsf{M}(d) \in \mathcal{O}(d^2)$. Furthermore, it follows from the algorithms in [Sch77, Nus80] that $\mathsf{M}(d)$ can be taken in $\mathcal{O}(d \log(d) \log(\log(d)))$.

In some situations, it may be helpful for better cost bounds to require the stronger bound $\mathsf{M}(d) \in \mathcal{O}(d^{\omega-1})$. It was done for example in [GSSV12] for the reduction of polynomial matrices and in [ZL12] for the computation of approximant bases. Since we assume $\omega > 2$ and there are known quasi-linear algorithms for polynomial multiplication, this is not an unreasonable assumption; yet, we will only use it sporadically, and always recall it explicitly.

In this thesis, we also use time functions related to the multiplication of univariate polynomial matrices. The multiplication time function $(m, d) \mapsto \mathsf{MM}(m, d)$ concerning $m \times m$ matrices of degree d is used for example in [Sto03, GJV03]. Like in these references, we also define a related quantity (see also Definition 12.1) that typically arises in divide-and-conquer computations.

Definition 6.1. *Let m and d be two positive real numbers. Then, $(m, d) \mapsto \mathsf{MM}(m, d)$ is such that two matrices of degree at most d in $\mathbb{K}[X]^{\bar{m} \times \bar{m}}$ with $\bar{m} \leqslant m$ can be multiplied using $\mathsf{MM}(m, d)$ operations in $\mathbb{K}$. Furthermore, we define*

$$\mathsf{MM}''(m, d) = \sum_{0 \leqslant i \leqslant \log(d)} 2^i \mathsf{MM}(m, 2^{-i} d).$$

It follows from [CK91] that one can always consider that $\mathsf{MM}(m,d) \in \mathcal{O}(m^\omega \mathsf{M}(d))$. In some situations, if the field $\mathbb{K}$ has sufficiently many elements, one may use an interpolation-evaluation schemes which yield better bounds such as $\mathcal{O}(m^\omega d + m^2 \mathsf{M}(d)\log(d))$; for more details, we refer to [Tho01, Tho02, BS05]. Here, unless indicated, we will not make such assumptions on $\mathbb{K}$; we will only use the following assumption on $\mathsf{MM}(\cdot,\cdot)$ which extends the super-linearity in Eq. (6.1) to polynomial matrices:

$$\mathcal{H}_{\mathsf{MM}(\cdot,\cdot)} \ : \quad \mathsf{MM}(m,d) + \mathsf{MM}(m,d') \leqslant \mathsf{MM}(m,d+d') \ \text{ for any } m,d,d' > 0. \tag{6.2}$$

It follows from this assumption that we have the upper bound

$$\mathsf{MM}''(m,d) \in \mathcal{O}(\mathsf{MM}(m,d)\log(d)) \ \subseteq \ \mathcal{O}(m^\omega \mathsf{M}(d)\log(d)). \tag{6.3}$$

We note that, in the case where one chooses for $\mathsf{MM}(m,d)$ some constant multiple of $m^\omega \mathsf{M}(d)$ or of $m^\omega d + m^2 \mathsf{M}(d)\log(d)$, then $\mathcal{H}_{\mathsf{MM}(\cdot,\cdot)}$ is a direct consequence of $\mathcal{H}_{\mathsf{M}(\cdot)}$.

6.2 Using the minimal degree to ensure uniform shift and output degrees

In this section, we consider the general case of univariate relation basis computation. For an arbitrary instance of Problem 4, we explain how the knowledge of the shifted minimal degree of the output module can be exploited in order to transform this instance into another one for which the input shift is almost uniform and any shifted minimal basis is known to have almost uniform degrees. This reduction is used in our fast algorithms for approximant bases (Chapter 7), for solution bases (Chapter 8), and for interpolant bases (Chapter 14), and combines ideas from Section 1.2.1 with partial linearization techniques from [Sto06].

Let $\mathbf{M}$, $\mathbf{F}$, and $\mathbf{s}$ be the input of Problem 4. We study the situation where the $\mathbf{s}$-minimal degree $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$ of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ is known. In this context, we first recall from Lemma 1.26 that the relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ in $-\boldsymbol{\delta}$-Popov form is actually equal to that in $\mathbf{s}$-Popov form.

Furthermore, this result also states the two following interesting properties. First, from any $-\boldsymbol{\delta}$-minimal relation basis $\mathbf{R}$ of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$, the $\mathbf{s}$-Popov basis can be retrieved with a simple constant unimodular transformation: it is $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})^{-1}\mathbf{R}$.

Besides, $\mathbf{R}$ has column degree $\mathrm{cdeg}(\mathbf{R}) = \boldsymbol{\delta}$, precisely like the sought $\mathbf{s}$-Popov basis; thus the sum of the column degrees of $\mathbf{R}$ is at most D according to Lemma 2.10, and its size is in $\mathcal{O}(m^2 + mD)$. A remaining difficulty may be the unbalancedness of these column degrees: while the sum of the entries of $\boldsymbol{\delta}$ is at most D, the amplitude of $\boldsymbol{\delta}$ may be large, for example when $\boldsymbol{\delta} = (D, 0, \ldots, 0)$.

Let us study the unbalancedness of the shift $\mathbf{d} = -\boldsymbol{\delta}$ in terms of the quantities discussed in Section 1.2.2 concerning assumption $\mathcal{H}_{\mathbf{s},1}$ and $\mathcal{H}_{\mathbf{s},2}$. First, the sum $|\mathbf{d} - \min(\mathbf{d})|$ may be as large as $\Theta(mD)$. However, we have $|\max(\mathbf{d}) - \mathbf{d}| \leqslant |\boldsymbol{\delta}| \leqslant D$, so that $\mathbf{d}$ satisfies $\mathcal{H}_{\mathbf{s},2}$. Techniques relying on the partial linearization in [Sto06, Section 3] have been already used to deal with such shifts in specific situations, namely for the computation of approximant bases [ZL12, Algorithm 2].

Furthermore, here we know the column degrees $\boldsymbol{\delta}$ of the output basis. Then this partial linearization leads to transforming our instance $(\mathbf{M}, \mathbf{F}, \mathbf{s})$ into another one, with slightly larger dimension m but with an almost uniform shift and with the guarantee that the degrees in the output basis are almost uniform.

Lemma 6.2. *Let* $\mathbf{M} \in \mathbb{K}^{D\times D}$, $\mathbf{F} \in \mathbb{K}^{m\times D}$, *and* $\mathbf{s} \in \mathbb{Z}^m$, *and let* $\boldsymbol{\delta} = (\delta_1, \dots, \delta_m)$ *denote the* $\mathbf{s}$*-minimal degree of* $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

Then, let $\delta = \lceil D/m \rceil \geqslant 1$, *and for* $i \in \{1, \dots, m\}$ *write* $\delta_i = (\alpha_i - 1)\delta + \beta_i$ *with* $\alpha_i = \lceil \delta_i/\delta \rceil$ *and* $1 \leqslant \beta_i \leqslant \delta$ *if* $\delta_i > 0$, *and with* $\alpha_i = 1$ *and* $\beta_i = 0$ *if* $\delta_i = 0$. *Let further* $\widetilde{m} = \alpha_1 + \cdots + \alpha_m$, *and define* $\widetilde{\boldsymbol{\delta}} \in \mathbb{Z}^{\widetilde{m}}_{\geqslant 0}$ *as*

$$\widetilde{\boldsymbol{\delta}} = (\underbrace{\delta, \dots, \delta, \beta_1}_{\alpha_1}, \dots, \underbrace{\delta, \dots, \delta, \beta_m}_{\alpha_m}) \tag{6.4}$$

and the expansion-compression matrix $\mathcal{E} \in \mathbb{K}[X]^{\widetilde{m}\times m}$ *as*

$$\mathcal{E} = \begin{bmatrix} 1 & & \\ X^\delta & & \\ \vdots & & \\ X^{(\alpha_1-1)\delta} & & \\ & \ddots & \\ & & 1 \\ & & X^\delta \\ & & \vdots \\ & & X^{(\alpha_m-1)\delta} \end{bmatrix}. \tag{6.5}$$

Let $\mathbf{d} = -\widetilde{\boldsymbol{\delta}} \in \mathbb{Z}^{\widetilde{m}}$ *and* $\mathbf{R} \in \mathbb{K}[X]^{\widetilde{m}\times\widetilde{m}}$ *be a* $\mathbf{d}$*-minimal relation basis of* $\mathrm{Syz}_{\mathbf{M}}(\mathcal{E}\cdot\mathbf{F})$. *Then, the* $\mathbf{s}$*-Popov relation basis of* $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ *is the submatrix of* $\mathrm{lm}_{\mathbf{d}}(\mathbf{R})^{-1}\mathbf{R}\mathcal{E}$ *formed by its rows at indices* $\alpha_1 + \cdots + \alpha_i$ *for* $1 \leqslant i \leqslant m$.

Furthermore, we have $m \leqslant \widetilde{m} \leqslant 2m$ *and* $\max(\mathbf{d}) - \min(\mathbf{d}) \leqslant \lceil D/m \rceil$.

Proof. We start by proving the inequalities on $\widetilde{m}$ and the amplitude of $\mathbf{d}$. We have $\alpha_i \leqslant 1 + \delta_i/\delta \leqslant 1 + m\delta_i/D$ for all i. Thus, since $|\boldsymbol{\delta}| \leqslant D$ according to Lemma 2.10, we obtain $\widetilde{m} = \alpha_1 + \cdots + \alpha_m \leqslant m + \sum_{1\leqslant i\leqslant m} m\delta_i/D \leqslant 2m$. The remark on $\mathbf{d}$ follows from the fact that all its entries are in $\{-\delta, \dots, 0\}$ with $\delta = \lceil D/m \rceil$.

Let $\mathbf{P}$ denote the $\mathbf{s}$-Popov relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$; $\mathbf{P}$ has column degree $\boldsymbol{\delta}$. First, we partially linearize the columns of $\mathbf{P}$ in degree δ to obtain $\overline{\mathbf{P}} \in \mathbb{K}[X]^{m\times\widetilde{m}}$; more precisely, $\overline{\mathbf{P}}$ is a matrix of degree at most δ and such that $\mathbf{P} = \overline{\mathbf{P}}\mathcal{E}$. Then, we define $\widetilde{\mathbf{P}} \in \mathbb{K}[X]^{\widetilde{m}\times\widetilde{m}}$ as follows:

- for $1 \leqslant i \leqslant m$, the row $\alpha_1 + \cdots + \alpha_i$ of $\widetilde{\mathbf{P}}$ is the row i of $\overline{\mathbf{P}}$;
- for $0 \leqslant i \leqslant m-1$ and $1 \leqslant j \leqslant \alpha_{i+1} - 1$, the row $\alpha_1 + \cdots + \alpha_i + j$ of $\widetilde{\mathbf{P}}$ is the row $[0, \cdots, 0, X^\delta, -1, 0, \cdots, 0] \in \mathbb{K}[X]^{1\times\widetilde{m}}$ with the entry X^δ at column index $\alpha_1 + \cdots + \alpha_i + j$.

Since $\mathbf{P}$ is in $\mathbf{s}$-Popov form with column degree $\boldsymbol{\delta}$, it is in $-\boldsymbol{\delta}$-Popov form by Lemma 1.26. Then, one can check that $\widetilde{\mathbf{P}}$ is in $\mathbf{d}$-Popov form and has $\mathbf{d}$-row degree $(0, \ldots, 0)$.

By construction, every row of $\widetilde{\mathbf{P}}$ is a relation of $\mathrm{Syz}_{\mathbf{M}}(\mathcal{E} \cdot \mathbf{F})$. In particular, since $\mathbf{R}$ is a relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathcal{E} \cdot \mathbf{F})$, there is a matrix $\mathbf{U} \in \mathbb{K}[X]^{\tilde{m} \times \tilde{m}}$ such that $\widetilde{\mathbf{P}} = \mathbf{UR}$. Besides, there is no relation $\widetilde{\mathbf{p}} \in \mathbb{K}[X]^{1 \times \tilde{m}}$ for $\mathrm{Syz}_{\mathbf{M}}(\mathcal{E} \cdot \mathbf{F})$ which has $\mathbf{d}$-degree less than 0: otherwise, $\widetilde{\mathbf{p}}\mathcal{E}$ would be a relation of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$, and it is easily checked that it would have $-\boldsymbol{\delta}$-degree less than 0, which is impossible.

Thus every row of $\mathbf{R}$ has $\mathbf{d}$-degree at least 0, and the predictable degree property in Theorem 1.11 shows that $\mathbf{U}$ is a constant matrix, and therefore unimodular. Then, $\widetilde{\mathbf{P}}$ is a relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathcal{E} \cdot \mathbf{F})$, and since it is in $\mathbf{d}$-Popov form, by Lemma 1.26 we obtain that $\widetilde{\mathbf{P}} = \mathrm{lm}_{\mathbf{d}}(\mathbf{R})^{-1}\mathbf{R}$. The conclusion follows. ☙

6.3 Computing residuals for systems of linear modular equations

In the framework of relation bases, the case of linear modular systems corresponds to a multiplication matrix $\mathbf{M}$ in companion-block diagonal form; our fast algorithms to solve Problem 4 in this case use divide-and-conquer approaches sketched in Sections 2.3 and 2.5. They involve in particular the computation of *residuals* which, for matrices $\mathbf{P} \in \mathbb{K}[X]^{m \times m}$ and $\mathbf{E} \in \mathbb{K}^{m \times D}$, are products of the form $\mathbf{P} \cdot \mathbf{E}$ as defined in Section 2.1.2. We showed in Section 2.5.1 that the multiplication matrix induces moduli $\mathfrak{M} = (\mathfrak{m}_1, \ldots, \mathfrak{m}_n) \in \mathbb{K}[X]^n_{\neq 0}$ and $\mathbf{E}$ induces a matrix $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ which are such that $\mathbf{P} \cdot \mathbf{E}$ corresponds to the modular polynomial matrix product $\mathbf{PF} \bmod \mathfrak{M}$, meaning that the column j is computed modulo $\mathfrak{m}_j$. In this section, we focus on the efficient computation of such residuals.

Let us describe the degree profiles of $\mathbf{P}$ and $\mathbf{F}$. The columns of $\mathbf{F}$ are considered modulo $\mathfrak{M}$, so that its column j has degree less than $\deg(\mathfrak{m}_j)$. Since we make no assumption on the degrees of the moduli, the column degrees of $\mathbf{F}$ may be unbalanced: all we know is a bound on their sum, $|\mathrm{cdeg}(\mathbf{F})| < \deg(\mathfrak{m}_1) + \cdots + \deg(\mathfrak{m}_n) = D$. On the other hand, in our algorithms, the matrix $\mathbf{P}$ is the outcome of a previous recursive call, and therefore is in shifted Popov form. Thus, we have a control over the degree of its determinant via Lemma 2.10, or in other words, over the sum of its column degrees, which will typically be close to D. To summarize, a good parameter to model the degrees of $\mathbf{P}$ and $\mathbf{F}$ is given by an integer $d \in \mathbb{Z}_{>0}$ for which $|\mathrm{cdeg}(\mathbf{P})| \leqslant d$ and $|\mathrm{cdeg}(\mathbf{F})| \leqslant d$.

We compute such a modular product $\mathbf{PF} \bmod \mathfrak{M}$ with unbalanced column degrees in two steps. The first one involves a partial linearization of the columns of $\mathbf{P}$, following ideas similar to those used in Lemma 6.2. This essentially helps us to reduce to the case of $\mathbf{P}$ having almost uniform column degrees, that is, $\deg(\mathbf{P}) \in \mathcal{O}(d/m)$.

Having such degrees for $\mathbf{P}$, in the second step, it remains to multiply $\mathbf{P}$ and $\mathbf{F}$, knowing that $|\mathrm{cdeg}(\mathbf{F})| \leqslant d$. For this, we also use a partial linearization technique, which is a simplified version of the one in [GSSV12, Section 6] for the purpose of multiplication. This technique consists in expanding the high-degree columns of $\mathbf{F}$ so as to obtain a matrix $\hat{\mathbf{F}}$ with more columns but degree at most d/m, then computing the product $\mathbf{P}\hat{\mathbf{F}}$, and finally retrieving the actual product $\mathbf{PF}$ by grouping together the columns that have

been expanded. Formally, we have the following result.

Lemma 6.3. *Let* $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ *and* $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$. *Let* $d \in \mathbb{Z}_{>0}$ *be such that* $d \geqslant m$, $\deg(\mathbf{P}) \leqslant d/m$, *and* $|\mathrm{cdeg}(\mathbf{F})| \leqslant d$. *Then, assuming* $n \in \mathcal{O}(m)$, *the product* $\mathbf{PF}$ *can be computed using* $\mathcal{O}(\mathsf{MM}(m, d/m))$ *operations in* $\mathbb{K}$.

Proof. We write $(d_1, \ldots, d_n) = \mathrm{cdeg}(\mathbf{F})$, and we let $\delta = \lfloor d/m \rfloor$. For $1 \leqslant j \leqslant n$, the column $\mathbf{F}_{*,j}$ of degree d_j is expanded into $\alpha_j = 1 + \lfloor d_j/(\delta+1) \rfloor$ columns $\hat{\mathbf{F}}_{*,(j,0)}, \ldots, \hat{\mathbf{F}}_{*,(j,\alpha_j-1)}$ each of degree at most δ, related by the identity

$$\mathbf{F}_{*,j} = \hat{\mathbf{F}}_{*,(j,0)} + X^{\delta+1}\hat{\mathbf{F}}_{*,(j,1)} + \cdots + X^{(\alpha_j-1)(\delta+1)}\hat{\mathbf{F}}_{*,(j,\alpha_j-1)}. \tag{6.6}$$

The expanded matrix $\hat{\mathbf{F}}$ has $\alpha_1 + \cdots + \alpha_n \leqslant n + \frac{d}{\delta+1} \leqslant n + m$ columns $\hat{\mathbf{F}}_{*,(j,i)}$. Assuming $n \in \mathcal{O}(m)$, the product $\mathbf{P}\hat{\mathbf{F}}$ can then be computed using $\mathcal{O}(\mathsf{MM}(m, d/m))$ operations in $\mathbb{K}$. It remains to perform the inverse operation, called partial compression: the column j of the product $\mathbf{PF}$ is obtained from the columns $(j, 0), \ldots, (j, \alpha_j - 1)$ of $\mathbf{P}\hat{\mathbf{F}}$ using the formula in Eq. (6.6). ☙

We now state our result concerning the computation of residuals. As in the lemma above, for efficiency we require that n be not much larger than m.

Lemma 6.4. *Let* $\mathfrak{M} = (\mathfrak{m}_j)_j \in \mathbb{K}[X]^n_{\neq 0}$ *and define* $\mathbf{D} = (D_j)_j \in \mathbb{Z}^n_{\geqslant 0}$ *by* $D_j = \deg(\mathfrak{m}_j)$ *for* $1 \leqslant j \leqslant n$. *Let* $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$, $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ *with* $\mathrm{cdeg}(\mathbf{F}) < \mathbf{D}$, *and let* $d \geqslant m$ *be such that* $d \geqslant D_1 + \cdots + D_n$ *and* $|\mathrm{cdeg}(\mathbf{P})| \leqslant d$. *Then, assuming* $n \in \mathcal{O}(m)$, *the modular product* $\mathbf{PF} \bmod \mathfrak{M}$ *can be computed in* $\mathcal{O}(\mathsf{MM}(m, d/m) + m\mathsf{M}(d))$ *operations.*

Proof. We start by using the partial linearization technique in Lemma 6.2 to make the column degrees of $\mathbf{P}$ almost uniform.

We use Lemma 6.2 with $\boldsymbol{\delta} = \mathrm{cdeg}(\mathbf{P})$ and $\delta = \lceil d/m \rceil$ to build a matrix $\mathcal{E} \in \mathbb{K}[X]^{\widetilde{m}\times m}$ as in Eq. (6.5); since $|\boldsymbol{\delta}| \leqslant d$, we have $m \leqslant \widetilde{m} \leqslant 2m$. Then, the columns of $\mathbf{P}$ can be expanded into a matrix $\widetilde{\mathbf{P}} \in \mathbb{K}[X]^{m\times\widetilde{m}}$ of smaller degree; precisely, it is such that $\mathbf{P} = \widetilde{\mathbf{P}}\mathcal{E}$ and $\deg(\widetilde{\mathbf{P}}) \leqslant \lceil d/m \rceil$. We note that $\mathcal{E}$ and $\widetilde{\mathbf{P}}$ are obtained without field operation.

Let us then define $\widetilde{\mathbf{F}} = \mathcal{E}\mathbf{F} \bmod \mathfrak{M}$. We will first compute $\widetilde{\mathbf{F}}$, and we will see that after that the sought modular product can be computed efficiently by relying on the identity $\mathbf{PF} \bmod \mathfrak{M} = \widetilde{\mathbf{P}}\,\widetilde{\mathbf{F}} \bmod \mathfrak{M}$.

We use notation from Lemma 6.2, denoting by α_i the number of powers of X in the column i of $\mathcal{E}$, for $1 \leqslant i \leqslant m$. We recall that $\mathcal{E}$ has $\widetilde{m} = \alpha_1 + \cdots + \alpha_m \leqslant 2m$ rows. Now, we write $\mathbf{F} = [f_{ij}]_{ij}$ and we let $j \in \{1, \ldots, n\}$. The column $\widetilde{\mathbf{F}}_{*,j} = \mathcal{E}\mathbf{F}_{*,j} \bmod \mathfrak{m}_j$ is formed by the m subcolumns $[X^{k\delta} f_{ij} \bmod \mathfrak{m}_j]^{\mathsf{T}}_{0\leqslant k<\alpha_i}$ for each $i \in \{1, \ldots, m\}$, where $\delta = \lceil d/m \rceil$ as above. The entries of the i-th subcolumn are computed iteratively in a total of $\mathcal{O}(\alpha_i(\mathsf{M}(D_j)+\mathsf{M}(d/m)))$ operations using fast polynomial division with remainder [GG13, Chapter 9]. With the super-linearity property $\mathcal{H}_{\mathsf{M}(\cdot)}$, summing these costs over i and j implies that $\widetilde{\mathbf{F}}$ can be computed in $\mathcal{O}((m+n)\mathsf{M}(d)) \subseteq \mathcal{O}(m\mathsf{M}(d))$ operations under the assumption $n \in \mathcal{O}(m)$.

Now, we have $\deg(\widetilde{\mathbf{P}}) \leqslant 2d/m$, as well as $\mathrm{cdeg}(\widetilde{\mathbf{F}}) < \mathbf{D}$ which implies $|\mathrm{cdeg}(\widetilde{\mathbf{F}})| < d$. Then, according to Lemma 6.3, the product $\widetilde{\mathbf{P}}\widetilde{\mathbf{F}}$ can be computed in $\mathcal{O}(\mathsf{MM}(m, d/m))$

field operations, assuming $n \in \mathcal{O}(m)$. The j-th column of $\widetilde{\mathbf{P}}\,\widetilde{\mathbf{F}}$ has $\widetilde{m} \leqslant 2m$ entries, each of degree at most $D_j + 2d/m$: this column can be reduced modulo $\mathfrak{m}_j$ in $\mathcal{O}(m(\mathsf{M}(D_j) + \mathsf{M}(d/m)))$ operations [GG13, Chapter 9]. Summing over $1 \leqslant j \leqslant n$, and assuming $n \in \mathcal{O}(m)$, this is in $\mathcal{O}((m+n)\mathsf{M}(d)) \subseteq \mathcal{O}(m\mathsf{M}(d))$. ☙

6.4 Computing shifted Popov relation bases iteratively for a triangular multiplication matrix [BL00]

In this section, we present in detail the iterative algorithm of [BL00, Algorithm FFFG], designed to compute relation bases as in Problem 4 when the multiplication matrix is upper triangular. The focus in [BL00] is on the design of a fraction-free algorithm, and therefore the cost bounds announced in this reference are pessimistic if one wants to apply this algorithm in a context such as ours here.

Below, we translate this algorithm into our framework, and we study its complexity in two cases: when the multiplication matrix is upper triangular, and more specifically when it is in Jordan form. The main difference between these two cases is that multiplying the Jordan matrix by a vector is linear in the dimension, while for a triangular matrix it is quadratic in general. If one has some motivations leading to consider intermediate situations between Jordan and triangular, then one may easily generalize the cost analysis so as to make the cost bound depend on the cost of vector-matrix products.

Proposition 6.5. *Algorithm 8 is correct and uses $\mathcal{O}(mD^2 + D^3)$ operations in $\mathbb{K}$. If the multiplication matrix $\mathbf{M}$ is in Jordan form, then it uses $\mathcal{O}(mD^2)$ operations in $\mathbb{K}$.*

Proof. Let us denote by $\mathbf{M}^{(j)} \in \mathbb{K}^{j\times j}$ the leading principal $j \times j$ submatrix of $\mathbf{M}$, and by $\mathbf{F}^{(j)} \in \mathbb{K}^{m\times j}$ the submatrix of $\mathbf{F}$ formed by its first j columns. To show the correctness of the algorithm, we prove the following loop invariant: at the end of the iteration j, the matrix $\mathbf{P}$ is the $\mathbf{s}$-Popov relation basis of $\mathrm{Syz}_{\mathbf{M}^{(j)}}(\mathbf{F}^{(j)})$, and we have $\mathbf{t} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})$ as well as $\mathbf{G} = \mathbf{P} \cdot \mathbf{F}$.

Concerning the initialization at Step **1**, the identity matrix is the $\mathbf{s}$-Popov basis of the whole module $\mathbb{K}[X]^{1\times m}$, $\mathbf{t} = \mathbf{s} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{I}_m)$, and $\mathbf{G} = \mathbf{F} = \mathbf{I}_m \cdot \mathbf{F}$.

Then, let $j \geqslant 1$ and assume that at the beginning of iteration j, the matrix $\mathbf{P}$ is the $\mathbf{s}$-Popov relation basis of $\mathrm{Syz}_{\mathbf{M}^{(j-1)}}(\mathbf{F}^{(j-1)})$, $\mathbf{t} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})$, and $\mathbf{G} = \mathbf{P} \cdot \mathbf{F}$. The fact that $\mathbf{M}$ is upper triangular implies that we have solved a first part of the problem: indeed, $\mathbf{P}$ being the $\mathbf{s}$-Popov relation basis of $\mathrm{Syz}_{\mathbf{M}^{(j-1)}}(\mathbf{F}^{(j-1)})$, it is precisely the $\mathbf{s}$-Popov basis of the module

$$\mathcal{M}^{(1)} = \{\mathbf{p} \in \mathbb{K}[X]^{1\times m} \mid \text{ the } j-1 \text{ first columns of } \mathbf{p}\cdot\mathbf{F} \text{ are zero}\}.$$

Aiming now at having the $\mathbf{s}$-Popov relation basis of $\mathrm{Syz}_{\mathbf{M}^{(j)}}(\mathbf{F}^{(j)})$, we follow the strategy indicated in Theorem 1.28, which leads us to consider the module

$$\begin{aligned}\mathcal{M}^{(2)} &= \{\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m} \mid \boldsymbol{\lambda}\mathbf{P} \text{ is a relation of } \mathrm{Syz}_{\mathbf{M}^{(j)}}(\mathbf{F}^{(j)})\}\\ &= \{\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m} \mid \text{the first } j \text{ columns of } (\boldsymbol{\lambda}\mathbf{P})\cdot\mathbf{F} \text{ are zero}\}\\ &= \{\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m} \mid \text{the column } j \text{ of } \boldsymbol{\lambda}\cdot\mathbf{G} \text{ is zero}\}.\end{aligned}$$

Algorithm 8 – ITERRELBAS
(Iterative relation basis for triangular mult. mat.)

Input:

- matrix $\mathbf{M} \in \mathbb{K}^{D\times D}$ upper triangular,
- matrix $\mathbf{F} \in \mathbb{K}^{m\times D}$,
- shift $\mathbf{s} \in \mathbb{Z}^m$.

Output:

- the $\mathbf{s}$-Popov relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

1. $\mathbf{P} \leftarrow \mathbf{I}_m$; $\mathbf{t} = (t_1, \ldots, t_m) \leftarrow \mathsf{copy}(\mathbf{s})$; $\mathbf{G} = [g_{ij}]_{i,j} \leftarrow \mathsf{copy}(\mathbf{F})$

2. $(x_1, \ldots, x_D) \leftarrow$ the diagonal entries of $\mathbf{M}$

3. For j from 1 to D:

If $\mathbf{G}_{*,j} \neq \mathbf{0}$:

a. $\pi \leftarrow \min\{i \in \{1, \ldots, m\} \mid g_{i,j} \neq 0 \text{ and } t_i = \min(\mathbf{t})\}$

b. $\mathbf{f}^{(1)} \leftarrow \dfrac{-1}{g_{\pi,j}} \begin{bmatrix} g_{1,j} \\ \vdots \\ g_{\pi-1,j} \end{bmatrix}$ and $\mathbf{f}^{(2)} \leftarrow \dfrac{-1}{g_{\pi,j}} \begin{bmatrix} g_{\pi+1,j} \\ \vdots \\ g_{m,j} \end{bmatrix}$

c. $\mathbf{P}^{(j)} \leftarrow \begin{bmatrix} \mathbf{I}_{\pi-1} & \mathbf{f}^{(1)} & \\ & X - x_j & \\ & \mathbf{f}^{(2)} & \mathbf{I}_{m-\pi} \end{bmatrix}$

d. $\mathbf{P} \leftarrow \mathbf{P}^{(j)}\mathbf{P}$

e. compute the residual $\mathbf{G} \leftarrow \mathbf{P}^{(j)} \cdot \mathbf{G}$ as follows:

- $\mathbf{G} \leftarrow \begin{bmatrix} \mathbf{I}_{\pi-1} & \mathbf{f}^{(1)} & \\ & 1 & \\ & \mathbf{f}^{(2)} & \mathbf{I}_{m-\pi} \end{bmatrix} \mathbf{G}$
- $\mathbf{G}_{\pi,*} \leftarrow \mathbf{G}_{\pi,*}(\mathbf{M} - x_j\mathbf{I}_D)$

f. $\mathbf{t} \leftarrow \mathrm{rdeg}_{\mathbf{t}}(\mathbf{P}^{(j)}) = (t_1, \ldots, t_{\pi-1}, t_\pi + 1, t_{\pi+1}, \ldots, t_m)$

g. $\boldsymbol{\delta} \leftarrow \mathbf{t} - \mathbf{s}$ with componentwise addition

h. $\begin{bmatrix} \mathbf{p}^{(1)} & 1 & \mathbf{p}^{(2)} \end{bmatrix} \leftarrow$ row π of $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{P})$

i. $\mathbf{U}^{(j)} \leftarrow \mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{P})^{-1} = \begin{bmatrix} \mathbf{I}_{\pi-1} & & \\ -\mathbf{p}^{(1)} & 1 & -\mathbf{p}^{(2)} \\ & & \mathbf{I}_{m-\pi} \end{bmatrix}$

j. $\mathbf{P} \leftarrow \mathbf{U}^{(j)}\mathbf{P}$; $\mathbf{G} \leftarrow \mathbf{U}^{(j)}\mathbf{G}$

4. Return $\mathbf{P}$

The last identity holds because the first $j-1$ columns of $\mathbf{P}\cdot\mathbf{F} = \mathbf{G}$ are zero by assumption, and thus since $\mathbf{M}$ is upper triangular the first $j-1$ columns of $\boldsymbol{\lambda}\cdot\mathbf{G}$ are zero for any $\boldsymbol{\lambda}$.

If the column $\mathbf{G}_{*,j}$ is zero, then the $\mathbf{t}$-Popov basis of $\mathcal{M}^{(2)}$ is the identity matrix; without changing $\mathbf{P}$, $\mathbf{t}$, and $\mathbf{G}$ we directly obtain that $\mathbf{P}$ is the $\mathbf{s}$-Popov relation basis of $\mathrm{Syz}_{\mathbf{M}^{(j)}}(\mathbf{F}^{(j)})$. In this case, the iteration j of Algorithm 8 does not perform any operation.

Now suppose that $\mathbf{G}_{*,j} \neq 0$, and let us use the notation π, $\mathbf{f}^{(1)}$, $\mathbf{f}^{(2)}$, and $\mathbf{P}^{(j)}$ from the algorithm. It is easily verified that $\mathbf{P}^{(j)}$ is the $\mathbf{t}$-Popov basis of $\mathcal{M}^{(2)}$. (Notice that the specific choice of π plays a role in this property: choosing another π such that $t_\pi = \min(\mathbf{t})$, if one exists, leads to a basis $\mathbf{P}^{(j)}$ which is $\mathbf{t}$-reduced but not $\mathbf{t}$-Popov.)

As a consequence, according to the item (iii) of Theorem 1.28, the product $\mathbf{P} \leftarrow \mathbf{P}^{(j)}\mathbf{P}$ is an $\mathbf{s}$-ordered weak Popov relation basis of $\mathrm{Syz}_{\mathbf{M}^{(j)}}(\mathbf{F}^{(j)})$. It remains to normalize this updated $\mathbf{P}$ into $\mathbf{s}$-Popov form.

At Step **3.f**, we compute $\mathbf{t} \leftarrow \mathrm{rdeg}_{\mathbf{t}}(\mathbf{P}^{(j)})$ which is thus $\mathbf{t} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})$. Then, the tuple $\boldsymbol{\delta}$ computed at Step **3.g** is the $\mathbf{s}$-pivot degree of $\mathbf{P}$, and thus the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_{\mathbf{M}^{(j)}}(\mathbf{F}^{(j)})$ by Lemma 1.25. By construction of $\mathbf{P}$, this matrix has column degree $\boldsymbol{\delta}$ and its $-\boldsymbol{\delta}$-leading matrix has the special shape

$$\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{P}) = \begin{bmatrix} \mathbf{I}_{\pi-1} & & \\ -\mathbf{p}^{(1)} & 1 & -\mathbf{p}^{(2)} \\ & & \mathbf{I}_{m-\pi} \end{bmatrix},$$

which comes from the fact that all rows of $\mathbf{P}$ except the row π are already normalized. In particular, $\mathbf{P}$ is in $-\boldsymbol{\delta}$-reduced form, and thus from Lemma 1.26 it follows that its $\mathbf{s}$-Popov form is equal to $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{P})^{-1}\mathbf{P}$. These remarks explain how we compute the $\mathbf{s}$-Popov basis in Steps **3.h** to **3.j**.

Finally, since the combination of Steps **3.e** and **3.j** computes $\mathbf{G} \leftarrow (\mathbf{U}^{(j)}\mathbf{P}^{(j)})\cdot\mathbf{G}$, we have $\mathbf{G} = \mathbf{P}\cdot\mathbf{F}$.

Let us now focus on the cost bound. We consider the iteration j of the For loop.

First, computing $\mathbf{P}^{(j)}$ and $\mathbf{U}^{(j)}$ involves $\mathcal{O}(m)$ field operations. Indeed, $\mathbf{P}^{(j)}$ is computed at Step **3.b** with less than m divisions, while for $\mathbf{U}^{(j)}$ one simply has to read the coefficients $[\mathbf{p}^{(1)}\ 1\ \mathbf{p}^{(2)}]$ of degree $\boldsymbol{\delta}$ of the row $\mathbf{P}_{\pi,*}$ (Step **3.h**), and to compute their opposites (Step **3.i**).

Concerning the residual $\mathbf{G} \in \mathbb{K}^{m\times D}$, the first multiplication at Step **3.e** and the one at Step **3.j** use both $\mathcal{O}(mD)$ field operations thanks to the specific form of the left-operands. Then, the vector-matrix product $\mathbf{G}_{\pi,*}(\mathbf{M} - x_j\mathbf{I}_D)$ at Step **3.e** uses $\mathcal{O}(D^2)$ operations in general, and $\mathcal{O}(mD)$ operations if $\mathbf{M}$ is in Jordan form, since in the latter case $\mathbf{M} - x_j\mathbf{I}_D$ has at most two nonzero elements per column.

It remains to study the computations related to $\mathbf{P}$ at Steps **3.d** and **3.j**. At the beginning of the iteration j, $\mathbf{P}$ is the $\mathbf{s}$-Popov relation basis of $\mathrm{Syz}_{\mathbf{M}^{(j-1)}}(\mathbf{F}^{(j-1)})$ and is thus such that $|\mathrm{cdeg}(\mathbf{P})| \leqslant j-1$ by Lemma 2.10. Then, Step **3.d** can be efficiently done by first left-multiplying the row π of $\mathbf{P}$ by the column π of $\mathbf{P}^{(j)}$, and then adding the other rows; that is, we use the formula

$$\mathbf{P} \leftarrow \mathbf{P}^{(j)}\mathbf{P} = \begin{bmatrix} \mathbf{f}^{(1)} \\ X - x_j \\ \mathbf{f}^{(2)} \end{bmatrix} \mathbf{P}_{\pi,*} + \begin{bmatrix} \mathbf{I}_{\pi-1} & & \\ & 0 & \\ & & \mathbf{I}_{m-\pi} \end{bmatrix} \mathbf{P}.$$

Since $|\mathrm{cdeg}(\mathbf{P})| \leqslant j-1$ and $\mathbf{P}$ is in $\mathbf{s}$-Popov form, it has at most $j-1$ columns which are not coordinate column vectors; as a result, $\mathbf{P}^{(j)}\mathbf{P}$ is computed in $\mathcal{O}(mj)$ field operations. Furthermore, we have explained above that the obtained $\mathbf{P}$ after this operation has sum of column degrees at most j, and although it is not necessarily in $\mathbf{s}$-Popov form, only its row π is not normalized. Then, similarly, the product $[-\mathbf{p}^{(1)}\ 1\ \ -\mathbf{p}^{(2)}]\mathbf{P}$ can be computed in $\mathcal{O}(mj)$ operations, hence an overall cost of $\mathcal{O}(mj) \subseteq \mathcal{O}(mD)$ operations for the computations related to $\mathbf{P}$ at iteration j.

Altogether, the iteration j uses $\mathcal{O}(mD + D^2)$ operations in general, and $\mathcal{O}(mD)$ for a Jordan matrix $\mathbf{M}$. Summing over $1 \leqslant j \leqslant D$ leads to the announced cost bound. ☙

We note that the cost bound still holds when D is small compared to m, although the output is an $m \times m$ matrix. The reason is that in this case, at least $m - D$ columns of the output basis $\mathbf{P}$ are coordinate vectors, which thus do not play a role in the cost of the computation.

For a triangular matrix, this cost bound $\mathcal{O}(mD^2 + D^3)$ can be compared to the cost $\mathcal{O}(mD^{\omega-1} + D^{\omega}\log(D))$ of our algorithm based on fast $\mathbb{K}$-linear algebra, presented in Section 2.2 and Chapter 4. The latter supports an arbitrary multiplication matrix, and is more efficient as soon as one uses sub-cubic matrix multiplication, that is, $\omega < 3$.

However, the iterative algorithm above manages to use the sparsity of the multiplication matrix, as long as it is triangular, obtaining for example the cost bound $\mathcal{O}(mD^2)$ in the case of a Jordan matrix. When $D > m$, our general algorithm works in $\mathcal{O}(D^{\omega}\log(D))$, which is slower than $\mathcal{O}(mD^2)$. Furthermore, this iteration can be turned into an efficient divide-and-conquer approach, as we develop in Chapter 14.

We note that, both in the iterative algorithm above and in our general algorithm of Chapter 4, computing $\mathbf{P}$ in $\mathbf{s}$-Popov form is an important ingredient to obtain an efficient algorithm since it gives us control over the column degrees of the output basis. For a nilpotent Jordan multiplication matrix, we will see in Example 7.5 that if one runs the iterative algorithm above without normalizing the bases at each iteration, then the sum of the column degrees of the output may be in $\Theta(mD)$. Thus, the size of the manipulated bases may be in $\Theta(m^2D)$, and over the D iterations this situation leads to a cost bound of $\mathcal{O}(m^2D^2)$ operations.

7

Computing shifted Popov approximant bases

In this chapter, we give fast algorithms to efficiently solve the general case of Problem 7, with arbitrary orders $\mathbf{D} = (D_1, \ldots, D_n)$ and an arbitrary shift $\mathbf{s}$. Furthermore, we put emphasis on computing the unique approximant basis in $\mathbf{s}$-Popov form.

A first solution to this problem is given by the quadratic iterative algorithms of [VBB91, BL94]. This approach was extended to solve Problem 8 in [VBB92] and then to solve Problem 4 with a triangular multiplication matrix in [BL00], the algorithm in the latter reference returning the shifted Popov basis. We presented this iterative solution in Section 6.4, with the following result (Proposition 6.5): there is an algorithm which solves Problem 7 in $\mathcal{O}(mD^2)$ operations in $\mathbb{K}$ and returns the basis in shifted Popov form, where $D = D_1 + \cdots + D_n$.

As soon as D is large compared to m, this is faster than the general $\mathcal{O}(D^\omega \log(D))$ algorithm of Part II. Computing the basis in shifted Popov form is at the core of the efficiency of this approach, by allowing us to control the degree growth in the bases that are manipulated in the process. The iterative algorithms in [VBB92, BL94] do not resort to normalized bases, and as a result their worst case cost bound is $\mathcal{O}(m^2D^2)$; we show such a worst case in Example 7.5 where the manipulated bases have size $\mathcal{O}(m^2D)$.

Further improvements are obtained by incorporating fast $\mathbb{K}$-linear algebra techniques and fast polynomial matrix multiplication into this iterative solution. Most previous results along this line focus on the case where all equations are modulo the same power of the variable, that is, Problem 7 with $D_1 = \cdots = D_n = D/n$; for a global overview of previous algorithms for approximant basis computation, we refer the reader to Section 2.3. We first study this situation in Section 7.1, presenting the algorithms of [BL94, GJV03] which are very efficient when $m \in \mathcal{O}(n)$. Then, for arbitrary orders $D_1, \ldots, D_n$, we show in Section 7.2 how these algorithms can be used to efficiently reduce the case $m \in \mathcal{O}(n)$ to the case $n \in \mathcal{O}(m)$.

In the latter situation, when n is small compared to m, a major difficulty is to control the degrees in the computed bases: they may be too large if the shift has a large amplitude, and they may be unbalanced for any shift. Assuming that the shift is close to uniform and that $D_1 = \cdots = D_n$, a fast solution was designed in [Sto06, ZL12] (see Proposition 2.16), controlling the unbalanced degrees by resorting to partial linearization techniques.

In Section 7.3, we use such techniques to give an efficient solution to the general case of Problem 7 when we have a priori information on the shifted minimal degree of the module of approximants. This chapter culminates with Section 7.4 where we design a divide-and-conquer algorithm to find this minimal degree, yielding a fast algorithm for Problem 7 for arbitrary shifts, arbitrary orders, and returning the shifted Popov basis.

7.1 Fast algorithms for almost uniform orders [GJV03]

In this section, we focus on the case $D_1 = \cdots = D_n = D/n$; for simplicity we denote this order by d and we write *approximant basis of* $\mathrm{Syz}_d(\mathbf{F})$ to refer to an approximant basis of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$ with $\mathbf{D} = (d, \ldots, d) \in \mathbb{Z}^n_{>0}$. Of course, a fast algorithm for such identical orders can also be applied to efficiently solve situations with almost uniform orders, for example if all of $D_1, \ldots, D_n$ are within a factor 2 of $(D_1 + \cdots + D_n)/n$.

We remark that, working with the orders $(d, \ldots, d)$, we will only compute bases that have degree at most d. Thus, one can hope for a cost bound which is quasi-linear in d. This was first achieved by [BL94, Algorithm SPHPS] which uses $\mathcal{O}(m^\omega \mathsf{M}(nd) \log(nd))$ field operations, and starts by transforming an instance of Problem 7 with n equations at order d to an instance with a single equation at order nd. For $n > 1$ equations, this algorithm was then improved in [GJV03, PM-Basis] by avoiding this transformation and incorporating fast $\mathbb{K}$-linear algebra to efficiently solve the case of n equations at order 1, which is the base case of the recursion. We now describe in details the approximant basis algorithms of [GJV03], which rely on the following particular case of Theorem 1.28.

Lemma 7.1. *Let $d \in \mathbb{Z}_{>0}$, let $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ with $\deg(\mathbf{F}) < d$, and let $\mathbf{s} \in \mathbb{Z}^m$. For some integers $d^{(1)} \in \{0, \ldots, d-1\}$ and $d^{(2)} = d - d^{(1)}$, let $\mathbf{P}^{(1)} \in \mathbb{K}[X]^{m\times m}$ be an approximant basis of $\mathrm{Syz}_{d^{(1)}}(\mathbf{F} \bmod X^{d^{(1)}})$, let $\mathbf{G} = (X^{-d^{(1)}}\mathbf{P}^{(1)}\mathbf{F}) \bmod X^{d^{(2)}}$, and let $\mathbf{P}^{(2)} \in \mathbb{K}[X]^{m\times m}$ be an approximant basis of $\mathrm{Syz}_{d^{(2)}}(\mathbf{G})$. Then, $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$ is an approximant basis of $\mathrm{Syz}_d(\mathbf{F})$. Furthermore, defining $\mathbf{t} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}^{(1)})$, if $\mathbf{P}^{(1)}$ and $\mathbf{P}^{(2)}$ are $\mathbf{s}$- and $\mathbf{t}$-reduced then $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$ is $\mathbf{s}$-reduced; and if $\mathbf{P}^{(1)}$ and $\mathbf{P}^{(2)}$ are in $\mathbf{s}$- and $\mathbf{t}$-ordered weak Popov form, then $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$ is in $\mathbf{s}$-ordered weak Popov form.*

Note that in this result and hereafter, we use the convention that any vector is an approximant of $\mathrm{Syz}_0(\mathbf{F})$; or in other words, a matrix in $\mathbb{K}[X]^{m\times m}$ is an approximant basis of $\mathrm{Syz}_0(\mathbf{F})$ if and only if it is unimodular.

First, we present the algorithm [GJV03, MBasis]. It uses the above result iteratively with $d^{(1)} = 1$ and $d^{(2)} = d - 1$, that is, increasing the order d by one at each iteration.

The improvement over the quadratic iterative Algorithm 8 is obtained by incorporating fast $\mathbb{K}$-linear algebra at the base case, which amounts to computing an $\mathbf{s}$-minimal approximant basis for a constant matrix $\mathbf{F} \in \mathbb{K}^{m\times n}$ at order $d = 1$. We detail this in Algorithm 9 below, which is a modified version of [GL14, Algorithm 1] or of the base case of [GJV03, Algorithm M-Basis] to ensure that the output is in $\mathbf{s}$-Popov form.

We remark that this instance at order 1 can be interpreted as the computation of a relation basis as in Problem 4, with as input the zero multiplication matrix $\mathbf{M} = \mathbf{0} \in \mathbb{K}^{n\times n}$, the matrix $\mathbf{F} \in \mathbb{K}^{m\times n}$, and the shift $\mathbf{s}$. In particular, since the output is of degree 1 and

thus close to a constant matrix, this can be solved efficiently by linearizing the problem over $\mathbb{K}$ as in Part II. With this point of view, Algorithm 9 can be seen as a simplified version of Algorithm 3, by exploiting the fact that $\mathbf{M} = \mathbf{0}$.

To give an idea of this simplification, we remark that in this case the degree bound for linearization can be taken as $\beta = 1$; thus, the striped Krylov matrix of Definition 4.4 is essentially a row permutation of $\mathbf{F}$ to reflect the priority of the rows induced by the shift (see Section 4.1). Thus, to find the $\mathbf{s}$-Popov approximant basis, the main computational task is to find the row and column rank profiles of this permuted $\mathbf{F}$, which uses $\mathcal{O}(\rho^{\omega-2}mn)$ operations with ρ the rank of $\mathbf{F}$ [Sto00, Section 2.2].

Algorithm 9 – LinAppBas
(Approximant basis with identical orders: base case [GJV03])

Input:
- scalar matrix $\mathbf{F} \in \mathbb{K}^{m\times n}$,
- shift $\mathbf{s} \in \mathbb{Z}^m$.

Output: the $\mathbf{s}$-Popov approximant basis of $\mathrm{Syz}_1(\mathbf{F})$.

1. $\pi_{\mathbf{s}} \leftarrow m \times m$ permutation matrix such that $[(s_1, 1) \ \cdots \ (s_m, m)]\pi_{\mathbf{s}}$ is lexicographically increasing
2. $(i_1, \ldots, i_\rho), (j_1, \ldots, j_\rho) \leftarrow$ row and column rank profiles of $\pi_{\mathbf{s}}\mathbf{F}$
3. $(k_1, \ldots, k_{m-\rho}) \leftarrow \{1, \ldots, m\} - \{i_1, \ldots, i_\rho\}$ sorted increasingly
4. $\mathbf{F}_1 \leftarrow$ submatrix of $\pi_{\mathbf{s}}\mathbf{F}$ with indices in $\{i_1, \ldots, i_\rho\} \times \{j_1, \ldots, j_\rho\}$
5. $\mathbf{F}_2 \leftarrow$ submatrix of $\pi_{\mathbf{s}}\mathbf{F}$ with indices in $\{k_1, \ldots, k_{m-\rho}\} \times \{j_1, \ldots, j_\rho\}$
6. $\pi \leftarrow$ permutation such that $[i_1 \ \cdots \ i_\rho \ k_1 \ \cdots \ k_{m-\rho}]\pi = [1 \ \cdots \ m]$
7. Return $\pi_{\mathbf{s}}^{-1}\pi^{-1} \begin{bmatrix} X\mathbf{I}_\rho & \mathbf{0} \\ -\mathbf{F}_2\mathbf{F}_1^{-1} & \mathbf{I}_{m-\rho} \end{bmatrix} \pi\pi_{\mathbf{s}}$

Proposition 7.2. *Algorithm 9 is correct and uses $\mathcal{O}(\rho^{\omega-2}mn)$ operations in $\mathbb{K}$, where ρ is the rank of $\mathbf{F}$.*

Proof. Step **1** corresponds to ordering the rows of $\mathbf{F}$ to follow the pivoting strategy given by $\mathbf{s}$, as we did in the more general construction of the multi-Krylov matrix in Section 4.1.

In this proof, we use the notation

$$\hat{\mathbf{s}} = \mathbf{s}\pi_{\mathbf{s}}, \quad \hat{\mathbf{F}} = \pi_{\mathbf{s}}\mathbf{F}, \quad \text{and} \quad \hat{\mathbf{P}} = \pi^{-1} \begin{bmatrix} X\mathbf{I}_\rho & \mathbf{0} \\ -\mathbf{F}_2\mathbf{F}_1^{-1} & \mathbf{I}_{m-\rho} \end{bmatrix} \pi,$$

and we show that $\hat{\mathbf{P}}$ is in $\hat{\mathbf{s}}$-Popov form, that all its rows are approximants of $\mathrm{Syz}_1(\hat{\mathbf{F}})$, and that they generate the module of such approximants.

We obviously have that $[X\mathbf{I}_\rho \;\; \mathbf{0}]\,\pi\,\hat{\mathbf{F}} = 0 \bmod X$. Now, by definition of π we have

$$\pi \begin{bmatrix} 1 \\ \vdots \\ m \end{bmatrix} = \begin{bmatrix} i_1 \\ \vdots \\ i_\rho \\ k_1 \\ \vdots \\ k_{m-\rho} \end{bmatrix}, \text{ and therefore } \pi\hat{\mathbf{F}} = \begin{bmatrix} \mathbf{F}_1 \\ \mathbf{F}_2 \end{bmatrix}.$$

This gives $[-\mathbf{F}_2\mathbf{F}_1^{-1} \;\; \mathbf{I}_{m-\rho}]\,\pi\,\hat{\mathbf{F}} = 0$, and thus all rows of $\hat{\mathbf{P}}$ are approximants of $\mathrm{Syz}_1(\hat{\mathbf{F}})$.

Besides, by definition of the row rank profile, the rightmost nonzero entry in the row i of $[-\mathbf{F}_2\mathbf{F}_1^{-1} \;\; \mathbf{I}_{m-\rho}]\,\pi$ is the 1 at index k_i; in other words, this matrix is a basis of the left nullspace of $\hat{\mathbf{F}}$ which is in a type of lower reduced echelon form. This implies that $\hat{\mathbf{P}}$ is lower triangular. Besides, $\hat{\mathbf{s}}$ is non-decreasing, the entries on the diagonal of $\hat{\mathbf{P}}$ are all 1 or X, and its non-diagonal entries are of degree less than the diagonal entry in the same column: $\hat{\mathbf{P}}$ is in $\hat{\mathbf{s}}$-Popov form.

The rows of $\pi_{\mathbf{s}}^{-1}\hat{\mathbf{P}}\pi_{\mathbf{s}}$ are approximants of $\mathrm{Syz}_1(\mathbf{F})$. Furthermore, since $\pi_{\mathbf{s}}$ performs a stable sort of $\mathbf{s}$, we have that $\pi_{\mathbf{s}}^{-1}\hat{\mathbf{P}}\pi_{\mathbf{s}}$ is in $\mathbf{s}$-Popov form. It remains to prove that this matrix generates the set of approximants of $\mathrm{Syz}_1(\mathbf{F})$. This follows from Lemma 1.8, since the degree of its determinant is $\deg(\det(\hat{\mathbf{P}})) = \rho$, which is also equal to the sum of the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_1(\mathbf{F})$ according to Corollary 4.11. ☙

Using this base case in an iterative fashion results in Algorithm 10, which is [GJV03, Algorithm M-Basis].

A cost bound for this algorithm is given below and roughly follows from the fact that an approximant basis at order d is an $m \times m$ matrix of degree at most d. Note that there is an additional term compared to the cost reported in [GJV03, Lemma 2.2] because this reference makes the assumption $n \leqslant m$, which we do not make here.

Proposition 7.3. *Algorithm 10 solves Problem 7 with $d = D_1 = \cdots = D_n = D/n$ using $\mathcal{O}(m^{\omega-1}(m+n)d^2)$ operations in $\mathbb{K}$ and returns a basis in $\mathbf{s}$-ordered weak Popov form.*

Proof. From Lemma 7.1 and Proposition 7.2, we have the following invariant: at the end of iteration j, $\mathbf{P}$ is an $\mathbf{s}$-ordered weak Popov approximant basis of $\mathrm{Syz}_j(\mathbf{F})$, and we have $\mathbf{G} = (X^{-j}\mathbf{P}\mathbf{F}) \bmod X^{d-j}$ and $\mathbf{t} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})$. Hence the correctness.

According to Proposition 7.2, Step **3.a** uses $\mathcal{O}(m^{\omega-1}n)$ operations in $\mathbb{K}$. At Step **3.b**, the product $\mathbf{P}^{(j)}\mathbf{P}$ involves two $m \times m$ matrices of degrees at most 1 and $d-1$, respectively, and hence can be computed in $\mathcal{O}(m^\omega d)$ operations. At Step **3.c**, one computes $\mathbf{P}^{(j)}\mathbf{G}$, with $\mathbf{P}^{(j)}$ a $m \times m$ matrix of degree at most 1 and $\mathbf{G}$ a $m \times n$ matrix of degree at most $d - j + 1$; this is done in $\mathcal{O}(m^{\omega-1}(m+n)d)$ operations. The announced cost follows. ☙

Comparing this to the cost $\mathcal{O}(m(nd)^2)$ from Proposition 6.5 concerning the iterative Algorithm 8, we conclude that Algorithm 10 is faster in some cases, including in particular when $m \in \mathcal{O}(n)$. This can be explained as follows: Algorithm 10 takes advantage of fast $\mathbb{K}$-linear algebra by working on $m \times n$ constant matrices at the base case but it ignores possibly unbalanced degrees in the output, while Algorithm 8 takes the average column

Algorithm 10 – IterAppBas
(Iterative approximant basis with identical orders [GJV03])

Input:

- positive integer $d \in \mathbb{Z}_{>0}$,
- matrix $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ of degree less than d,
- shift $\mathbf{s} \in \mathbb{Z}^m$.

Output: an $\mathbf{s}$-ordered weak Popov approximant basis of $\mathrm{Syz}_d(\mathbf{F})$.

1. $\mathbf{P} \leftarrow \mathbf{I}_m$ `// s-Popov basis of` $\mathrm{Syz}_0(\mathbf{F})$
2. $\mathbf{G} \leftarrow \mathrm{copy}(\mathbf{F})$; $\mathbf{t} \leftarrow \mathrm{copy}(\mathbf{s})$
3. For j from 1 to d:
   a. $\mathbf{P}^{(j)} \leftarrow \mathrm{LinAppBas}(\mathbf{G} \bmod X, \mathbf{t})$ `// t-Popov basis of` $\mathrm{Syz}_1(\mathbf{G})$
   b. $\mathbf{P} \leftarrow \mathbf{P}^{(j)}\mathbf{P}$ `// s-ordered weak Popov basis of` $\mathrm{Syz}_j(\mathbf{F})$
   c. $\mathbf{G} \leftarrow (X^{-1}\mathbf{P}^{(j)}\mathbf{G}) \bmod X^{d-j}$; $\mathbf{t} \leftarrow \mathrm{rdeg}_{\mathbf{t}}(\mathbf{P}^{(j)}) = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})$
4. Return $\mathbf{P}$

degree of the output into account but uses only naïve linear algebra. It turns out that the case $m \in \mathcal{O}(n)$ is particularly in favor of Algorithm 10 since it both minimizes the output degree unbalancedness and maximizes the efficiency of fast linear algebra.

Now, we study [GJV03, PM-Basis], which uses a divide-and-conquer approach, relying on Lemma 7.1 with $d^{(1)} = d^{(2)} = d/2$. This allows us to incorporate fast polynomial multiplication: instead of multiplying two matrices of respective degrees 1 and $d-1$, we will multiply two matrices of degree about $d/2$.

Proposition 7.4. *Algorithm 11 solves Problem 7 with $d = D_1 = \cdots = D_n = D/n$ using*

$$\mathcal{O}\Big(\Big(1+\frac{n}{m}\Big)\,\mathsf{MM}''(m,d)\Big) \subseteq \mathcal{O}\Big(\Big(1+\frac{n}{m}\Big)\,\mathsf{MM}(m,d)\log(d)\Big) \subseteq \mathcal{O}(m^{\omega-1}(m+n)\mathsf{M}(d)\log(d))$$

operations in $\mathbb{K}$ and returns a basis in $\mathbf{s}$-ordered weak Popov form.

Proof. First, the basis computed at Step **1** is in shifted Popov form according to Proposition 7.2. Then, the facts that Algorithm 11 is correct and that it returns a basis in $\mathbf{s}$-ordered weak Popov form both follow from Lemma 7.1.

For the cost analysis, we assume that d is a power of 2. As explained above, the base case of the recursion at Step **1** uses $\mathcal{O}(m^{\omega-1}n)$ operations, which is in $\mathcal{O}(\frac{n}{m}\mathsf{MM}(m,1))$. Then, we have two recursive calls at Steps **2.b** and **2.d**, with the same dimension m and at order $d/2$.

The residual $\mathbf{G}$ at Step **2.c** is obtained from the product $\mathbf{P}^{(1)}\mathbf{F}$, where $\mathbf{P}^{(1)}$ is an $m \times m$ matrix of degree at most $d/2$, and $\mathbf{F}$ is an $m \times n$ matrix of degree at most d. This product

Algorithm 11 – DACAPPBAS
(Divide-and-conquer app. basis with identical orders [GJV03])

Input:

- positive integer $d \in \mathbb{Z}_{>0}$,
- matrix $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ of degree less than d,
- shift $\mathbf{s} \in \mathbb{Z}^m$.

Output: an $\mathbf{s}$-ordered weak Popov approximant basis $\mathbf{P}$ for $\mathrm{Syz}_d(\mathbf{F})$.

1. If $d = 1$ then return LINAPPBAS$(\mathbf{F}, \mathbf{s})$

2. Else:

a. $d^{(1)} \leftarrow \lceil d/2 \rceil$, $d^{(2)} \leftarrow d - d^{(1)}$

b. $\mathbf{P}^{(1)} \leftarrow$ DACAPPBAS$(d^{(1)}, \mathbf{F} \bmod X^{d^{(1)}}, \mathbf{s})$

c. $\mathbf{G} \leftarrow (X^{-d^{(1)}} \mathbf{P}^{(1)} \mathbf{F}) \bmod X^{d^{(2)}}$; $\mathbf{t} \leftarrow \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}^{(1)})$

d. $\mathbf{P}^{(2)} \leftarrow$ DACAPPBAS$(d^{(2)}, \mathbf{G}, \mathbf{t})$

e. Return $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$

is done in $\mathcal{O}(\mathsf{MM}(m, d))$ operations if $n \leqslant m$, and in $\mathcal{O}(\frac{n}{m}\mathsf{MM}(m, d))$ operations if $m \leqslant n$. The multiplication at Step **2.e** involves two $m \times m$ matrices of degree at most $d/2$, and hence is done in $\mathcal{O}(\mathsf{MM}(m, d))$ operations in $\mathbb{K}$. Using Eq. (6.3), the announced bound follows. ☙

From Lemma 2.10, the degree of the determinant of an approximant basis of $\mathrm{Syz}_d(\mathbf{F})$ is at most $D = nd$, and the discussion in Section 1.2.2 indicates that we can set $\tilde{\mathcal{O}}(m^{\omega-1}nd)$ as a target cost. This target is achieved by Algorithm 11 when $m \in \mathcal{O}(n)$. However, the situation is less satisfactory when n is negligible compared to m, and in particular for $n = 1$. Indeed, in the latter case, Algorithm 11 costs $\tilde{\mathcal{O}}(m^{\omega}d)$ operations while there is hope to achieve $\tilde{\mathcal{O}}(m^{\omega-1}d)$.

In the next sections, we will design algorithms that will eventually allow us to reach this target cost bound $\tilde{\mathcal{O}}(m^{\omega-1}D)$ in Section 7.4 for Problem 7 in its full generality. Previously, similarly fast algorithms had only been provided for identical orders and for shifts that are close to uniform [Sto06, ZL12], as summarized in Proposition 2.16.

The main difficulty that arises when dealing with small n and arbitrary shifts is that, although we have control over the degree of the determinant of an approximant basis, we lose control over the actual degrees of its entries which are not anymore uniformly distributed even for bases in shifted ordered weak Popov form. We now highlight this through an example where the size of the approximant basis computed by Algorithm 10 is beyond our target cost. We also illustrate the fact, explained in Section 1.2.2, that considering the more specific shifted Popov basis gives us control over the average column degree; this will thus be a central idea in our algorithms.

Example 7.5. We focus on a Hermite-Padé approximation problem, that is, $n = 1$. We

consider a specific input $\mathbf{F}$ of dimension $2m \times 1$ described below, an order $D \in \mathbb{Z}_{>0}$ with $D \geqslant m$, and the shift $\mathbf{s} = (0, \ldots, 0, D, \ldots, D) \in \mathbb{Z}_{\geqslant 0}^{2m}$ with m entries 0 and m entries D.

Let f be a polynomial in X with nonzero constant coefficient, and let $f_1, \ldots, f_m$ be generic polynomials in X of degree less than D. Then, we consider the following vector with all entries truncated modulo X^D:

$$\mathbf{F} = \begin{bmatrix} f \\ f + Xf \\ X(f + Xf) \\ \vdots \\ X^{m-2}(f + Xf) \\ f_1 \\ \vdots \\ f_m \end{bmatrix}.$$

After m loop iterations, Algorithm 10 has computed an $\mathbf{s}$-ordered weak Popov basis $\mathbf{P}^{(m)}$ of approximant of $\mathrm{Syz}_m(\mathbf{F})$, which is such that $\mathbf{t} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}^{(m)}) = (1, \ldots, 1, D, \ldots, D)$ and

$$\mathbf{P}^{(m)}\mathbf{F} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ X^m f \\ X^m g_1 \\ \vdots \\ X^m g_m \end{bmatrix} \bmod X^D$$

for some polynomials $g_1, \ldots, g_m$.

Now, continuing the loop until reaching order D, since the coefficient of degree m of $X^m f$ is nonzero and because of the specific shift $\mathbf{t}$, the obtained approximant basis $\mathbf{P}$ of $\mathrm{Syz}_D(\mathbf{F})$ has degree profile

$$\mathbf{P} = \begin{bmatrix} [1] & [0] & & & & & \\ \vdots & \ddots & \ddots & & & & \\ [1] & \cdots & [1] & [0] & & & \\ [\delta+1] & \cdots & [\delta+1] & [\delta+1] & & & \\ [\delta] & \cdots & [\delta] & [\delta] & [0] & & \\ \vdots & \cdots & \vdots & \vdots & & \ddots & \\ [\delta] & \cdots & [\delta] & [\delta] & & & [0] \end{bmatrix},$$

where $\delta = D - m$, $[i]$ denotes an entry of degree i, the entries left blank correspond to the zero polynomial, and the entries $[\delta + 1]$ are on the row m of $\mathbf{P}$. In particular, $\mathbf{P}$ has size $\Theta(m^2 D)$. Note that $\mathbf{P}$ is in $\mathbf{s}$-ordered weak Popov form; however, the $\mathbf{s}$-Popov

approximant basis of $\mathrm{Syz}_D(\mathbf{F})$ has degrees

$$\begin{bmatrix} [1] & [0] & & & & & \\ \vdots & \ddots & \ddots & & & & \\ [0] & \cdots & [1] & [0] & & & \\ [0] & \cdots & [0] & [\delta+1] & & & \\ [0] & \cdots & [0] & [\delta] & [0] & & \\ \vdots & \cdots & \vdots & \vdots & & \ddots & \\ [0] & \cdots & [0] & [\delta] & & & [0] \end{bmatrix};$$

in particular, this basis has average column degree D/m and size in $\mathcal{O}(mD)$. ✍

7.2 Arbitrary orders: reduction to the case $n \in \mathcal{O}(m)$

Here, as a first building block of our fast algorithm, we deal with the case of $n > m$ equations with arbitrary orders, relying on the above divide-and-conquer Algorithm 11. More precisely, we give an efficient reduction, which partly solves the approximation equations and leaves us with an approximant basis problem with less than m equations.

Let us consider an input for Problem 7: we have positive integers $\mathbf{D} = (D_1, \ldots, D_n)$, a matrix $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ with $\mathrm{cdeg}(\mathbf{F}) < \mathbf{D}$ componentwise, and a shift $\mathbf{s} \in \mathbb{Z}^m$. We let $D = |\mathbf{D}| = D_1 + \cdots + D_n$ and we assume that $n > m$.

First, for ease of presentation, we reduce to the case where all orders are powers of two. This can be done by considering for each j the power of two immediately larger than D_j, giving us new orders $\tilde{\mathbf{D}} = (\tilde{D}_1, \ldots, \tilde{D}_n)$ defined as $\tilde{D}_j = 2^{\lceil \log_2(D_j) \rceil}$ for $1 \leqslant j \leqslant n$. Denoting by $\tilde{\mathbf{D}} - \mathbf{D}$ the componentwise difference, the matrix $\tilde{\mathbf{F}} = \mathbf{F}\mathbf{X}^{\tilde{\mathbf{D}}-\mathbf{D}}$ has column degree bounded by $\tilde{\mathbf{D}}$ componentwise, and it is easily verified that the set of approximant of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$ is equal to the set of approximant of $\mathrm{Syz}_{\tilde{\mathbf{D}}}(\tilde{\mathbf{F}})$. In this process, the sum of orders does not incur more than a twofold increase: $\tilde{D} = |\tilde{\mathbf{D}}| \leqslant 2D$, which is an important fact with regard to complexity bounds.

Besides, up to reordering the equations, we assume that $\tilde{D}_1 \geqslant \cdots \geqslant \tilde{D}_n$ without loss of generality. Then denoting by ℓ the integer such that $\tilde{D}_m = 2^\ell$, we define

$$\nu_i = \mathrm{Card}(\{j \in \{1, \ldots, n\} \mid \tilde{D}_j = 2^i\})$$

for $0 \leqslant i \leqslant \ell$, as well as $\nu_{\ell+1} = n - \nu_0 - \cdots - \nu_\ell$. This means that

$$(\tilde{D}_1, \ldots, \tilde{D}_n) = (\underbrace{> 2^\ell, \ldots, > 2^\ell}_{\nu_{\ell+1}}, \underbrace{2^\ell, \ldots, 2^\ell}_{\nu_\ell}, \underbrace{2^{\ell-1}, \ldots, 2^{\ell-1}}_{\nu_{\ell-1}}, \ldots, \underbrace{1, \ldots, 1}_{\nu_0}). \tag{7.1}$$

We also define $\mu_i = \nu_{\ell+1} + \nu_\ell + \cdots + \nu_i = \max\{j \mid \tilde{D}_j = 2^i\}$ for all $0 \leqslant i \leqslant \ell$, which is the number of equations whose associated order is at least 2^i.

Let us sketch our algorithm, based on this decomposition of $\tilde{\mathbf{D}}$. It will first compute a basis $\mathbf{P}^{(0)}$ for all $\mu_0 = n$ equations at order $2^0 = 1$. After this step, we are left with the residual equations $\mathbf{G} = X^{-1}\mathbf{P}^{(0)}\mathbf{F}$ and the associated orders $\tilde{D}_1 - 1, \ldots, \tilde{D}_n - 1$; the last ν_0 orders are zero. Thus, we continue by computing an approximant basis $\mathbf{P}^{(1)}$ for

these $\mu_1 = n - \nu_0$ equations at order $2^1 - 2^0 = 1$, giving an approximant basis $\mathbf{P}^{(1)}\mathbf{P}^{(0)}$ for $\mathbf{F}$ at order $2^1 = 2$. Then we compute an approximant basis $\mathbf{P}^{(2)}$ at order $2^2 - 2^1 = 2$ for the μ_2 residual equations given by $\mathbf{G} = X^{-2}\mathbf{P}^{(1)}\mathbf{P}^{(0)}\mathbf{F}$, giving the approximant basis $\mathbf{P}^{(2)}\mathbf{P}^{(1)}\mathbf{P}^{(0)}$ for $\mathbf{F}$ at order $2^2 = 4$. Continuing this process iteratively until reaching the order 2^ℓ, we are finally left with $\nu_{\ell+1} = \mu_{\ell+1}$ equations, with $\mu_{\ell+1} < m$.

This procedure is formalized in Algorithm 12, with a cost bound and correctness given in the next proposition. Besides, we remark that in this context with $n \geqslant m$, we have $D = D_1 + \cdots + D_n \geqslant m$ as well.

Algorithm 12 – ReduceNbEqAppBas
(Approximant basis: reduction to $n \in \mathcal{O}(m)$)

Input:

- positive integers $\mathbf{D} = (D_1, \ldots, D_n) \in \mathbb{Z}^n_{>0}$ with $D_1 \geqslant \cdots \geqslant D_n$,
- matrix $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ with $\operatorname{cdeg}(\mathbf{F}) < \mathbf{D}$ componentwise and $n \geqslant m$,
- shift $\mathbf{s} \in \mathbb{Z}^m$.

Output:

- $\mathbf{P}$ an $\mathbf{s}$-ordered weak Popov approximant basis at order $\mathbf{D} - (\hat{\mathbf{D}}, 0, \ldots, 0)$ for $\mathbf{F}$,
- $\hat{\mathbf{D}} = (D_1 - D_m, \ldots, D_\nu - D_m) \in \mathbb{Z}^\nu_{>0}$, where $\nu = \max\{j \mid D_j > D_m\}$,
- $\hat{\mathbf{F}} = X^{-D_m}[(\mathbf{P}\mathbf{F}_{*,1}) \bmod X^{D_1} | \cdots | (\mathbf{P}\mathbf{F}_{*,\nu}) \bmod X^{D_\nu}] \in \mathbb{K}[X]^{m\times\nu}$,
- $\hat{\mathbf{s}} = \operatorname{rdeg}_{\mathbf{s}}(\mathbf{P}) \in \mathbb{Z}^m$.

1. $\tilde{D}_j \leftarrow 2^{\lceil \log_2(D_j) \rceil}$ for $m \leqslant j \leqslant n$
2. $\tilde{D}_j \leftarrow D_j + \tilde{D}_m - D_m$ for $1 \leqslant j < m$
3. $\ell \leftarrow \log_2(\tilde{D}_m)$
4. $\mu_i \leftarrow \max\{j \mid \tilde{D}_j \geqslant 2^i\}$ for $1 \leqslant i \leqslant \ell$
5. $\nu \leftarrow \max\{j \mid \tilde{D}_j > 2^\ell\}$
6. $\tilde{\mathbf{F}} \leftarrow \mathbf{F}\mathbf{X}^{\tilde{\mathbf{D}}-\mathbf{D}}$ where $\tilde{\mathbf{D}} = (\tilde{D}_1, \ldots, \tilde{D}_n)$
7. $\mathbf{P} \leftarrow$ LinAppBas$(\tilde{\mathbf{F}} \bmod X, \mathbf{s})$
8. For i from 1 to ℓ:
   a. $\mathbf{G} \leftarrow (X^{-2^{i-1}}\mathbf{P}[\tilde{\mathbf{F}}_{*,1} | \cdots | \tilde{\mathbf{F}}_{*,\mu_i}]) \bmod X^{2^{i-1}}$
   b. $\mathbf{P}^{(i)} \leftarrow$ DaCAppBas$(\mathbf{G}, 2^{i-1}, \operatorname{rdeg}_{\mathbf{s}}(\mathbf{P}))$
   c. $\mathbf{P} \leftarrow \mathbf{P}^{(i)}\mathbf{P}$
9. $\hat{\mathbf{D}} \leftarrow (D_1 - D_m, \ldots, D_\nu - D_m)$; $\hat{\mathbf{s}} \leftarrow \operatorname{rdeg}_{\mathbf{s}}(\mathbf{P})$
10. $\hat{\mathbf{F}} \leftarrow X^{-D_m}[(\mathbf{P}\mathbf{F}_{*,1}) \bmod X^{D_1} | \cdots | (\mathbf{P}\mathbf{F}_{*,\nu}) \bmod X^{D_\nu}]$
11. Return $(\mathbf{P}, \hat{\mathbf{D}}, \hat{\mathbf{F}}, \hat{\mathbf{s}})$

Proposition 7.6. *Algorithm 12 is correct and uses*

$$\mathcal{O}(\mathsf{MM}(m, D/m)\log(D/m)) \subseteq \mathcal{O}(m^{\omega}\mathsf{M}(D/m)\log(D/m))$$

operations in $\mathbb{K}$*, where* $D = D_1 + \cdots + D_m$*. Furthermore, the output* $(\mathbf{P}, \hat{\mathbf{D}}, \hat{\mathbf{F}}, \hat{\mathbf{s}})$ *is such that* $\hat{\mathbf{F}}$ *has* m *rows and* $\nu < m$ *columns,* $|\hat{\mathbf{D}}| \leqslant D$*,* $\hat{\mathbf{s}} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})$*,* $\deg(\mathbf{P}) \leqslant 2D/m$*, and for any approximant basis* $\mathbf{Q}$ *of* $\mathrm{Syz}_{\hat{\mathbf{D}}}(\hat{\mathbf{F}})$*, then* $\mathbf{QP}$ *is an approximant basis of* $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$*.*

Proof. Steps **1** to **6** compute the parameters described above as well as the matrix $\tilde{\mathbf{F}}$. Step **7** computes an $\mathbf{s}$-Popov approximant basis $\mathbf{P}$ at order $(1, \ldots, 1) \in \mathbb{Z}^n_{>0}$ for $\tilde{\mathbf{F}}$. Then, thanks to Theorem 1.28 we have the following loop invariant: at the end of iteration i, $\mathbf{P}$ is an $\mathbf{s}$-ordered weak Popov approximant basis at order $(2^i, \ldots, 2^i, \tilde{D}_{\mu_i+1}, \ldots, \tilde{D}_n)$ for $\tilde{\mathbf{F}}$.

Thus, when arriving at Step **9** the matrix $\mathbf{P}$ is an $\mathbf{s}$-ordered weak Popov approximant basis at order $(2^\ell, \ldots, 2^\ell, \tilde{D}_{\mu_\ell+1}, \ldots, \tilde{D}_n)$ for $\tilde{\mathbf{F}}$; since $\mu_\ell \geqslant m$, the latter tuple can also be written

$$(\tilde{D}_m, \ldots, \tilde{D}_m, \tilde{D}_{m+1}, \ldots, \tilde{D}_n) = \tilde{\mathbf{D}} - (\hat{\mathbf{D}}, 0, \ldots, 0)$$

where the difference is taken componentwise. By choice of $\tilde{\mathbf{F}} = \mathbf{F}\mathbf{X}^{\tilde{\mathbf{D}}-\mathbf{D}}$, we obtain that $\mathbf{P}$ is an approximant basis at order

$$\mathbf{D} - (\hat{\mathbf{D}}, 0, \ldots, 0) = (D_m, \ldots, D_m, D_{m+1}, \ldots, D_n)$$

for $\mathbf{F}$. In particular, the fact that $\mathbf{QP}$ is an approximant basis of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$ follows from Theorem 1.28.

Now, let us prove the cost bound. As detailed in Section 7.1, Step **7** uses $\mathcal{O}(m^{\omega-1}n)$ operations since $n \geqslant m$. Thanks to the super-linearity property $\mathcal{H}_{\mathsf{MM}(\cdot,\cdot)}$ of Eq. (6.2), this is in $\mathcal{O}(\frac{n}{m}\mathsf{MM}(m, 1)) \subseteq \mathcal{O}(\mathsf{MM}(m, n/m))$, and thus fits into the announced bound since $n \leqslant D$. The resulting approximant basis $\mathbf{P}$ has degree at most 1.

The computation at Step **8.a** can be done by first computing $\mathbf{P}[\tilde{\mathbf{F}}_{*,1}|\cdots|\tilde{\mathbf{F}}_{*,\mu_i}] \bmod X^{2^i}$. Since $\mathbf{F}$ is a $m \times \mu_i$ matrix with $\mu_i \geqslant m$, this product is done in $\mathcal{O}(\frac{\mu_i}{m}\mathsf{MM}(m, 2^i))$ operations. Then, according to Proposition 7.4, Step **8.b** uses $\mathcal{O}(\frac{\mu_i}{m}\mathsf{MM}(m, 2^{i-1})\log(2^{i-1}))$ operations. Furthermore, the output $\mathbf{P}^{(i)}$ has degree at most 2^{i-1}, and thus at Step **9.c** we multiply two $m \times m$ matrices of degree at most 2^{i-1}; this is done in $\mathcal{O}(\mathsf{MM}(m, 2^i))$ operations. Altogether, the loop at Step **8** uses

$$\begin{aligned}\mathcal{O}\left(\sum_{i=1}^{\ell}\frac{\mu_i}{m}\mathsf{MM}(m, 2^{i-1})\log(2^i)\right) &\subseteq \mathcal{O}\left(\mathsf{MM}\left(m, \sum_{i=1}^{\ell}\frac{\mu_i 2^{i-1}}{m}\right)\log(2^\ell)\right)\\ &\subseteq \mathcal{O}(\mathsf{MM}(m, D/m)\log(D/m))\end{aligned}$$

field operations. The first inclusion relies on the super-linearity property $\mathcal{H}_{\mathsf{MM}(\cdot,\cdot)}$ stated in Eq. (6.2), while the second inclusion follows from the following remarks. On the one hand, since $D \geqslant D_1 + \cdots + D_m$ and $D_1 \geqslant \cdots \geqslant D_m \geqslant \tilde{D}_m/2 = 2^{\ell-1}$, we have $2^\ell \leqslant 2D/m$ and thus $\log(2^\ell) \in \mathcal{O}(\log(D/m))$. On the other hand, as shown in Eq. (7.1), we have by construction

$$\begin{aligned}2D \geqslant |\tilde{\mathbf{D}}| &= \tilde{D}_1 + \cdots + \tilde{D}_\nu + \nu_\ell 2^\ell + \cdots + \nu_1 2^1 + \nu_0 2^0\\ &> \mu_\ell(2^\ell - 2^{\ell-1}) + \cdots + \mu_2(2^2 - 2^1) + \mu_1(2^1 - 2^0) + \mu_0 2^0,\end{aligned}$$

and thus in particular $\sum_{i=1}^{\ell} \mu_i 2^{i-1} \in \mathcal{O}(D)$.

The matrix $\hat{\mathbf{F}}$ at Step **11** can be directly obtained from the product $\mathbf{P}[\mathbf{F}_{*,1}|\cdots|\mathbf{F}_{*,\nu}]$. Since $\mathbf{P}$ has degree at most $2D/m$, and since these $\nu < m$ columns have degree less than $D_1, \ldots, D_\nu$ respectively, with $D_1 + \cdots + D_\nu < D$, this product can be computed in $\mathcal{O}(\mathsf{MM}(m, D/m))$ operations in $\mathbb{K}$ according to Lemma 6.3. ☙

7.3 Fast approximant bases in Popov form with known minimal degree

We now study the situation where we know a priori the shifted minimal degree of the considered module of approximants, or in other words, the diagonal degrees of the shifted Popov approximant basis. This additional information allows us to design a fast algorithm for the general case Problem 7, that is, with cost bound $\tilde{\mathcal{O}}(m^{\omega-1}D)$ for any dimensions m and n, for arbitrary orders $D_1, \ldots, D_n$, and arbitrary shift $\mathbf{s}$. Furthermore, our algorithm returns the $\mathbf{s}$-Popov approximant basis.

From Section 7.2, we know that the main remaining difficulty is to deal with $\mathbf{F}$ having fewer columns than rows. As noted in Section 7.1, this may result in unbalanced degrees in the computed bases. Such non-uniform degrees are commonly handled by resorting to partial linearization techniques; in this context, knowing the shifted minimal degree will help us to make these techniques particularly efficient.

We will use two types of partial linearization, adapted from those of [Sto06]. The first one was presented in Section 6.2 and ensures that the shift is almost uniform and that the output matrix has low degrees. Here we detail the second one, which takes advantage of the results of the first one to further transform the instance so as to ensure that the orders $D_1, \ldots, D_n$ are all of the same magnitude and that the dimensions m and n are roughly the same. Then, this obtained approximant basis problem can be solved efficiently with the divide-and-conquer algorithm designed for identical orders, namely Algorithm 11.

To give an overview of the main idea behind this reduction, let us consider the case of one column. The input is $D \in \mathbb{Z}_{>0}$, $\mathbf{F} \in \mathbb{K}[X]^{m\times 1}$ with $\deg(\mathbf{F}) < D$, $\mathbf{s} \in \mathbb{Z}^m$, and the $\mathbf{s}$-minimal degree $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$ of $\mathrm{Syz}_D(\mathbf{F})$. For our construction, we choose any integer $\delta \geqslant \max(\boldsymbol{\delta})$. Note that in general we may have $\max(\boldsymbol{\delta})$ as large as D, in which case the partial linearization below is useless; however, the underlying idea is that one will first apply the partial linearization of Section 6.2, which reduces to the case $\max(\boldsymbol{\delta}) \leqslant \lceil D/m \rceil$. Thus, one may read the next paragraphs bearing in mind that the integer we have chosen is such that $\delta \approx D/m$.

Then, a type of column partial linearization of $\mathbf{F}$ is defined in [Sto06] as follows. Writing $D = \alpha\delta + \beta$ with $\alpha = \left\lceil \frac{D}{\delta} - 1 \right\rceil$ and $1 \leqslant \beta \leqslant \delta$, we consider the X^δ-adic representation

$$\mathbf{F} = \mathbf{F}^{(0)} + \mathbf{F}^{(1)}X^\delta + \cdots + \mathbf{F}^{(\alpha)}X^{\alpha\delta} \text{ with } \deg(\mathbf{F}^{(j)}) < \delta \text{ for } j < \alpha \text{ and } \deg(\mathbf{F}^{(\alpha)}) < \beta,$$

and the expanded matrix

$$\mathcal{L}^{\mathrm{ab}}_{D,\delta}(\mathbf{F}) = \begin{bmatrix} \mathbf{F}^{(0)} + \mathbf{F}^{(1)}X^{\delta} & \mathbf{F}^{(1)} + \mathbf{F}^{(2)}X^{\delta} & \cdots & \mathbf{F}^{(\alpha-1)} + \mathbf{F}^{(\alpha)}X^{\delta} \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}.$$

The main point is that approximants of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$ are related to approximants at order $(2\delta, \ldots, 2\delta, \delta+\beta)$ for $\mathcal{L}^{\mathrm{ab}}_{D,\delta}(\mathbf{F})$. This construction thus gives us a new approximant basis problem with more equations and smaller orders: if $\delta \approx D/m$, we have $\alpha \approx m$ equations with associated orders $2\delta \approx 2D/m$. Furthermore, Proposition 7.4 shows that such an instance of Problem 7 can be solved efficiently, in $\tilde{\mathcal{O}}(m^{\omega-1}D)$ field operations.

We sketch the link between the approximants as follows. Let $\mathbf{p} \in \mathbb{K}[X]^{1\times m}$ be such that $\deg(\mathbf{p}) \leqslant \delta$ and $\mathbf{p}$ is an approximant of $\mathrm{Syz}_D(\mathbf{F})$. Then, by construction $[\mathbf{p} \;\; \mathbf{q}]$ is an approximant at order $(2\delta, \ldots, 2\delta, \delta+\beta)$ for $\mathcal{L}^{\mathrm{ab}}_{D,\delta}(\mathbf{F})$, where $\mathbf{q} = [q_1, \ldots, q_{\alpha-1}] \in \mathbb{K}[X]^{1\times\alpha-1}$ is defined by $q_j = X^{-j\delta}\mathbf{p}(\mathbf{F}^{(0)} + \cdots + \mathbf{F}^{(j-1)}X^{(j-1)\delta})$, or in other words, q_j is formed by the part of $X^{-\delta}\mathbf{p}\mathbf{F}^{(j-1)}$ of nonnegative degree. We remark that $\deg(\mathbf{q}) < \deg(\mathbf{p})$.

In fact, we are also interested is the converse property, since we would like to obtain an approximant basis of $\mathrm{Syz}_D(\mathbf{F})$ from the computation of an approximant basis at order $(2\delta, \ldots, 2\delta, \delta+\beta)$ for $\mathcal{L}^{\mathrm{ab}}_{D,\delta}(\mathbf{F})$. As seen in the paragraph above, degree constraints on the approximants play a role in the link between these two instances of Problem 7, and actually only some specific shifts will allow us to recover an approximant basis of $\mathrm{Syz}_D(\mathbf{F})$ from an approximant basis for the transformed problem. This is where the a priori knowledge of the shifted minimal degree of the sought approximant basis is brought into play: it precisely tells us how to choose such a shift.

We now give the details of this partial linearization in general; it was introduced in [Sto06, Section 2] and used previously in [GS11, ZL12]. In [Sto06, ZL12] one does not assume the knowledge of the shifted minimal degree, yet this transformation still succeeds thanks to assumptions on the shift which constrain the degrees in the approximant basis; namely, these are the assumptions $\mathcal{H}_{\mathbf{s},1}$ and $\mathcal{H}_{\mathbf{s},2}$ (see Section 1.2.2).

Definition 7.7. *Let $\mathbf{D} \in \mathbb{Z}^n_{>0}$, $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ with $\mathrm{cdeg}(\mathbf{F}) < \mathbf{D}$ and $n < m$, $\mathbf{s} \in \mathbb{Z}^m$, and let $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$ be the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$. Let further $\delta \in \mathbb{Z}_{>0}$ be such that $\delta \geqslant \max(\boldsymbol{\delta})$, and for $1 \leqslant i \leqslant n$, $D_i = \alpha_i\delta + \beta_i$ with $\alpha_i = \lceil \frac{D_i}{\delta} - 1 \rceil$ and $1 \leqslant \beta_i \leqslant \delta$. Writing the X^{δ}-adic representation*

$$\mathbf{F}_{*,i} = \mathbf{F}^{(0)}_{*,i} + \mathbf{F}^{(1)}_{*,i}X^{\delta} + \cdots + \mathbf{F}^{(\alpha_i)}_{*,i}X^{\alpha_i\delta} \text{ with } \deg(\mathbf{F}^{(j)}_{*,i}) < \delta \text{ for } j < \alpha_i \text{ and } \deg(\mathbf{F}^{(\alpha_i)}_{*,i}) < \beta_i,$$

we consider the expanded matrix

$$\widetilde{\mathbf{F}}_{*,i} = \begin{bmatrix} \mathbf{F}^{(0)}_{*,i} + \mathbf{F}^{(1)}_{*,i}X^{\delta} & \mathbf{F}^{(1)}_{*,i} + \mathbf{F}^{(2)}_{*,i}X^{\delta} & \cdots & \mathbf{F}^{(\alpha_i-1)}_{*,i} + \mathbf{F}^{(\alpha_i)}_{*,i}X^{\delta} \end{bmatrix} \in \mathbb{K}[X]^{m\times\alpha_i}$$

and the matrix $\mathcal{E}_i = \begin{bmatrix} \mathbf{0} & \mathbf{I}_{\alpha_i-1} \end{bmatrix} \in \mathbb{K}[X]^{\alpha_i-1\times\alpha_i}$ if $\alpha_i > 1$, and otherwise $\widetilde{\mathbf{F}}_{*,i} = \mathbf{F}_{*,i}$ and

$\mathcal{E}_i \in \mathbb{K}[X]^{0\times 1}$. *Then, we define the partial linearization of* $\mathbf{F}$ *as*

$$\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{F}) = \begin{bmatrix} \widetilde{\mathbf{F}}_{*,1} & \widetilde{\mathbf{F}}_{*,2} & \cdots & \widetilde{\mathbf{F}}_{*,n} \\ \mathcal{E}_1 & & & \\ & \mathcal{E}_2 & & \\ & & \ddots & \\ & & & \mathcal{E}_n \end{bmatrix} \in \mathbb{K}[X]^{(m+\widetilde{n})\times(n+\widetilde{n})},$$

where the number of additional columns is $\widetilde{n} = \max(\alpha_1 - 1, 0) + \cdots + \max(\alpha_n - 1, 0)$. *We also expand* D_i *as* $\widetilde{D}_i = (2\delta, \ldots, 2\delta, \delta + \beta_i) \in \mathbb{Z}^{\alpha_i}_{>0}$ *if* $\alpha_i > 1$ *and* $\widetilde{D}_i = D_i$ *otherwise, and we define* $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{D}) = (\widetilde{D}_1, \ldots, \widetilde{D}_n) \in \mathbb{Z}^{\widetilde{n}}_{>0}$.

This transformation only induces a moderate increase of the dimensions of the problem, as long as δ is not chosen too small. Indeed, we transform an $m \times n$ instance into an $(m + \widetilde{n}) \times (n + \widetilde{n})$ instance, where $\widetilde{n} \leqslant \alpha_1 + \cdots + \alpha_n < D/\delta$.

On the other hand, when $\boldsymbol{\delta}$ is known, we have seen in Section 6.2 how to ensure that $\max(\boldsymbol{\delta}) \leqslant \lceil 2D/m \rceil$ via partial linearization techniques. Thus, in this situation, we can choose $\delta = \lceil 2D/m \rceil$, and we have $\widetilde{n} < D/\delta \leqslant m/2$. While adding at most $m/2$ to the dimensions m and n, this transformation allows us to consider small and almost uniform orders since $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{D})$ has all its entries between 0 and $2\delta \in \mathcal{O}(D/m)$.

As a consequence, Algorithm 11 efficiently computes an approximant basis for the transformed instance $(\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{D}), \mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{F}))$. We now give a precise link between approximants for the latter instance, and approximants for the original instance $(\mathbf{D}, \mathbf{F})$.

Lemma 7.8. *Let* $\mathbf{D} \in \mathbb{Z}^n_{>0}$, $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ *with* $\mathrm{cdeg}(\mathbf{F}) < \mathbf{D}$ *and* $n < m$, $\mathbf{s} \in \mathbb{Z}^m$, $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$ *be the* $\mathbf{s}$*-minimal degree for* $(\mathbf{D}, \mathbf{F})$, *and* $\delta \geqslant \max(\boldsymbol{\delta})$. *Let* $\widetilde{\mathbf{P}}$ *be a* $(-\boldsymbol{\delta}, -\delta, \ldots, -\delta)$*-ordered weak Popov approximant basis at order* $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{D})$ *for* $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{F})$. *Then, the leading principal submatrix* $\mathbf{R} \in \mathbb{K}[X]^{m\times m}$ *of* $\widetilde{\mathbf{P}}$ *is a* $-\boldsymbol{\delta}$*-ordered weak Popov approximant basis of* $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$, *and* $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})^{-1}\mathbf{R}$ *is the* $\mathbf{s}$*-Popov approximant basis of* $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$.

Proof. We write $\mathbf{d}$ for the shift $\mathbf{d} = (-\boldsymbol{\delta}, -\delta, \ldots, -\delta) \in \mathbb{Z}^{m+\widetilde{n}}$.

Let $\mathbf{p} \in \mathbb{K}[X]^{1\times m}$ be an approximant of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$ such that $\mathrm{rdeg}_{-\boldsymbol{\delta}}(\mathbf{p}) = 0$; remark that all approximant of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$ have nonnegative $-\boldsymbol{\delta}$-degree (see the paragraph after Definition 1.22). We are going to show that this induces an approximant $[\mathbf{p} \;\; \mathbf{q}] \in \mathbb{K}[X]^{1\times(m+\widetilde{n})}$ at order $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{D})$ for $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{F})$ with $\deg(\mathbf{q}) < \deg(\mathbf{p})$ and thus $\mathrm{rdeg}_{\mathbf{d}}([\mathbf{p} \;\; \mathbf{q}]) = 0$.

First, we consider $i \in \{1, \ldots, n\}$ such that $\alpha_i \in \{0, 1\}$. Then, $\widetilde{\mathbf{F}}_{*,i} = \mathbf{F}_{*,i}$, $\widetilde{D}_i = D_i$, and $\mathcal{E}_i \in \mathbb{K}[X]^{0\times 1}$. Defining $\mathbf{q}_i$ as an empty vector in $\mathbb{K}[X]^{1\times 0}$, the identity $\mathbf{p}\mathbf{F}_{*,i} = 0 \bmod X^{D_i}$ can be rewritten as $\begin{bmatrix}\mathbf{p} & \mathbf{q}_i\end{bmatrix}\begin{bmatrix}\widetilde{\mathbf{F}}_{*,i} \\ \mathcal{E}_i\end{bmatrix} = 0 \bmod X^{\widetilde{D}_i}$. Now, we consider i such that $\alpha_i > 1$ and we define $\mathbf{q}_i = [q_{1,i}, \ldots, q_{\alpha_i-1,i}] \in \mathbb{K}[X]^{1\times(\alpha_i-1)}$ by

$$q_{j,i} = X^{-j\delta}\mathbf{p}(\mathbf{F}^{(0)}_{*,i} + \cdots + \mathbf{F}^{(j-1)}_{*,i}X^{(j-1)\delta}). \tag{7.2}$$

In other words, since $\deg(\mathbf{p}) \leqslant \delta$ and $\deg(\mathbf{F}^{(k)}_{*,i}) < \delta$ for all k, $q_{j,i}$ is formed by the part of $X^{-\delta}\mathbf{p}\mathbf{F}^{(j-1)}_{*,i}$ of nonnegative degree, and in particular $\deg(\mathbf{q}_i) < \deg(\mathbf{p})$. Then, we

have $\mathbf{p}(\mathbf{F}_{*,i}^{(0)} + \cdots + \mathbf{F}_{*,i}^{(j+1)} X^{(j+1)\delta}) = 0 \bmod X^{(j+2)\delta}$ for $j < \alpha_i - 1$, hence by construction $q_{j,i} X^{j\delta} + \mathbf{p}(\mathbf{F}_{*,i}^{(j)} X^{j\delta} + \mathbf{F}_{*,i}^{(j+1)} X^{(j+1)\delta}) = 0 \bmod X^{(j+2)\delta}$, and it immediately follows that $\mathbf{F}_{*,i}^{(j)} + \mathbf{F}_{*,i}^{(j+1)} X^{\delta} + q_{j,i} = 0 \bmod X^{2\delta}$. The same arguments for $j = \alpha_i - 1$ yield the identity $\mathbf{F}_{*,i}^{(\alpha_i - 1)} + \mathbf{F}_{*,i}^{(\alpha_i)} X^{\delta} + q_{j,i} = 0 \bmod X^{\delta + \beta_i}$. In short, we have

$$\begin{bmatrix} \mathbf{p} & \mathbf{q}_i \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{F}}_{*,i} \\ \mathcal{E}_i \end{bmatrix} = 0 \bmod (X^{2\delta}, \ldots, X^{2\delta}, X^{\delta+\beta_i}). \tag{7.3}$$

Thus, by construction of $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{F})$ and $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{D})$, this implies that $[\mathbf{p} \;\; \mathbf{q}]$ is an approximant at order $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{D})$ for $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{F})$, where $\mathbf{q} = [\mathbf{q}_1 \;\; \cdots \;\; \mathbf{q}_n]$ is such that $\deg(\mathbf{q}) < \deg(\mathbf{p})$.

More generally, let $\mathbf{P} \in \mathbb{K}[X]^{m \times m}$ be a $-\boldsymbol{\delta}$-ordered weak Popov approximant basis of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$; then $\mathrm{rdeg}_{-\boldsymbol{\delta}}(\mathbf{P}) = \mathbf{0}$ by Lemma 1.26. According to the above discussion, there is a matrix $\mathbf{Q} \in \mathbb{K}[X]^{m \times (n+\widetilde{n})}$ such that all rows of $[\mathbf{P} \;\; \mathbf{Q}]$ are approximants at order $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{D})$ for $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{F})$, and $\mathrm{rdeg}(\mathbf{Q}) < \mathrm{rdeg}(\mathbf{P})$. Then, $[\mathbf{P} \;\; \mathbf{Q}]$ is in $\mathbf{d}$-ordered weak Popov form with all $\mathbf{d}$-pivots in $\mathbf{P}$, since we have $\mathrm{lm}_{\mathbf{d}}([\mathbf{P} \;\; \mathbf{Q}]) = [\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{P}) \;\; \mathbf{0}]$ with $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{P})$ lower triangular by assumption.

Now, let us show conversely that if $[\mathbf{p} \;\; \mathbf{q}] \in \mathbb{K}[X]^{1 \times (m+\widetilde{n})}$ is an approximant at order $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{D})$ for $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{F})$ with $\deg(\mathbf{q}) < \delta$ and $\deg(\mathbf{p}) \leqslant \delta$, then $\mathbf{p}$ is an approximant of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$. Let us write $\mathbf{q} = [\mathbf{q}_1 \;\; \cdots \;\; \mathbf{q}_n]$ with $\mathbf{q}_i \in \mathbb{K}[X]^{1 \times 0}$ if $\alpha_i \in \{0, 1\}$ and $\mathbf{q}_i = [q_{1,i}, \ldots, q_{\alpha_i - 1, i}] \in \mathbb{K}[X]^{1 \times \alpha_i - 1}$ if $\alpha_i > 1$. Our goal is to prove that $\mathbf{p}\mathbf{F}_{*,i} = 0 \bmod X^{D_i}$ for all $i \in \{1, \ldots, n\}$; this is obvious for i such that $\alpha_i \in \{0, 1\}$.

Let $i \in \{1, \ldots, n\}$ be such that $\alpha_i > 1$, and let us consider the identities given by

$$\begin{bmatrix} \mathbf{p} & \mathbf{q}_i \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{F}}_{*,i} \\ \mathcal{E}_i \end{bmatrix} = 0 \bmod (X^{2\delta}, \ldots, X^{2\delta}, X^{\delta+\beta_i}).$$

The first column gives the relation $\mathbf{p}(\mathbf{F}_{*,i}^{(0)} + \mathbf{F}_{*,i}^{(1)} X^{\delta}) = 0 \bmod X^{2\delta}$, while the other columns yield $\mathbf{p}(\mathbf{F}_{*,i}^{(j)} + \mathbf{F}_{*,i}^{(j+1)} X^{\delta}) = -q_{j,i} \bmod X^{2\delta}$ for $1 \leqslant j \leqslant \alpha_i - 2$ and $\mathbf{p}(\mathbf{F}_{*,i}^{(\alpha_i - 1)} + \mathbf{F}_{*,i}^{(\alpha_i)} X^{\delta}) = -q_{\alpha_i - 1, i} \bmod X^{\delta + \beta_i}$. Using our assumption on $\deg(\mathbf{q})$ and $\deg(\mathbf{p})$, these identities for $j = 0$ and $j = 1$ imply that the terms of degree less than 2δ of $\mathbf{p}(\mathbf{F}_{*,i}^{(0)} + \mathbf{F}_{*,i}^{(1)} X^{\delta} + \mathbf{F}_{*,i}^{(2)} X^{2\delta})$ are $0 = \mathbf{p}\mathbf{F}_{*,i}^{(0)} - q_{1,i} X^{\delta}$. Therefore, we obtain both $q_{1,i} = X^{-\delta} \mathbf{F}_{*,i}^{(0)}$ and $\mathbf{p}\mathbf{F}_{*,i} = 0 \bmod X^{3\delta}$. This process can be continued with $j = 2$, up to $j = \alpha_i - 1$, to eventually obtain that $\mathbf{p}\mathbf{F}_{*,i} = 0 \bmod X^{D_i}$. (We remark that this also implies that $\mathbf{q}$ is given by $\mathbf{p}$ as in Eq. (7.2).)

To conclude the proof, let $\widetilde{\mathbf{P}} \in \mathbb{K}[X]^{(m+\widetilde{n}) \times (m+\widetilde{n})}$ be a $\mathbf{d}$-ordered weak Popov approximant basis at order $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{D})$ for $\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\mathbf{F})$. We write

$$\widetilde{\mathbf{P}} = \begin{bmatrix} \mathbf{R} & \widetilde{\mathbf{P}}^{(12)} \\ \widetilde{\mathbf{P}}^{(21)} & \widetilde{\mathbf{P}}^{(22)} \end{bmatrix} \text{ with } \mathbf{R} \in \mathbb{K}[X]^{m \times m} \text{ and } \widetilde{\mathbf{P}}^{(22)} \in \mathbb{K}[X]^{\widetilde{n} \times \widetilde{n}},$$

the other dimensions being defined implicitly.

First, since $\mathbf{R}$ contains the $\mathbf{d}$-pivots of $[\mathbf{R} \;\; \widetilde{\mathbf{P}}^{(12)}]$ and $\mathbf{d} = (-\boldsymbol{\delta}, -\delta, \ldots, -\delta)$, by minimality of $\widetilde{\mathbf{P}}$ we have that the $-\boldsymbol{\delta}$-pivot degree of $\mathbf{R}$ is at most $\boldsymbol{\delta}$ componentwise. This implies in particular that $\deg(\mathbf{R}) \leqslant \max(\boldsymbol{\delta}) = \delta$ and $\deg(\widetilde{\mathbf{P}}^{(12)}) < \delta$. Then, from the discussion above applied to each of the first m rows of $\widetilde{\mathbf{P}}$, we obtain that $\mathbf{R}\mathbf{F} = 0 \bmod \mathbf{X}^{\mathbf{D}}$.

As a consequence, by minimality of $\boldsymbol{\delta}$, the $-\boldsymbol{\delta}$-pivot degree of $\mathbf{R}$ is at least $\boldsymbol{\delta}$ componentwise, and hence is equal to $\boldsymbol{\delta}$. Thus, $\mathbf{R}$ is a $-\boldsymbol{\delta}$-ordered weak Popov approximant basis of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$. Then, according to Lemma 1.26, $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})^{-1}\mathbf{R}$ is the $\mathbf{s}$-Popov approximant basis of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$. ☙

Algorithm 13 – MinDegAppBas
(Approximant basis with known minimal degree)

Input:

- positive integers $\mathbf{D} = (D_1, \ldots, D_n) \in \mathbb{Z}^n_{>0}$,
- matrix $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ with $\mathrm{cdeg}(\mathbf{F}) < \mathbf{D}$ and $D_1 + \cdots + D_n > m$,
- shift $\mathbf{s} \in \mathbb{Z}^m$,
- the $\mathbf{s}$-minimal degree $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m) \in \mathbb{Z}^m_{\geqslant 0}$ of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$.

Output: the $\mathbf{s}$-Popov approximant basis of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$.

1. `/* Reduce to almost uniform shift and output degrees */`
   $\delta \leftarrow \lceil (D_1 + \cdots + D_n)/m \rceil$,
   $\alpha_i \leftarrow \max(1, \lceil \delta_i/\delta \rceil)$ for $1 \leqslant i \leqslant m$,
   $\widetilde{m} \leftarrow \alpha_1 + \cdots + \alpha_m$
   $\widetilde{\boldsymbol{\delta}} \in \mathbb{Z}^{\widetilde{m}}_{\geqslant 0}$ as in Eq. (6.4)
   $\mathcal{E} \in \mathbb{K}[X]^{\widetilde{m}\times m}$ as in Eq. (6.5) and $\widetilde{\mathbf{F}} \leftarrow \mathcal{E}\mathbf{F} \bmod \mathbf{X}^{\mathbf{D}}$
2. `/* Reduce to less equations than unknowns */`
   $(\mathbf{R}^{(1)}, \hat{\mathbf{D}}, \hat{\mathbf{F}}, -\hat{\boldsymbol{\delta}}) \leftarrow \begin{cases} \textsc{ReduceNbEqAppBas}(\mathbf{D}, \widetilde{\mathbf{F}}, -\widetilde{\boldsymbol{\delta}}) & \text{if } n \geqslant \widetilde{m} \\ (\mathbf{I}_{\widetilde{m}}, \mathbf{D}, \widetilde{\mathbf{F}}, -\widetilde{\boldsymbol{\delta}}) & \text{if } n < \widetilde{m} \end{cases}$
   $\nu \leftarrow$ the number of columns of $\hat{\mathbf{F}} \in \mathbb{K}[X]^{\widetilde{m}\times\nu}$ `//` $\nu < \widetilde{m}$
3. `/* Reduce to almost uniform orders */`
   $\mathcal{L}^{\mathrm{ab}}_{\hat{\mathbf{D}},\delta}(\hat{\mathbf{D}}) \in \mathbb{Z}^{\widetilde{m}+\widetilde{\nu}}_{>0}$ and $\mathcal{L}^{\mathrm{ab}}_{\hat{\mathbf{D}},\delta}(\hat{\mathbf{F}}) \in \mathbb{K}[X]^{(\widetilde{m}+\widetilde{\nu})\times(\nu+\widetilde{\nu})}$ as in Definition 7.7
   $\mathbf{d} \leftarrow (-\hat{\boldsymbol{\delta}}, -\delta, \ldots, -\delta)$
4. `/* Compute approximant basis */`
   $\hat{d} \leftarrow \max(\hat{\mathbf{D}})$; $\boldsymbol{\Delta} \leftarrow (\hat{d}, \ldots, \hat{d}) - \mathcal{L}^{\mathrm{ab}}_{\hat{\mathbf{D}},\delta}(\hat{\mathbf{D}})$
   $\widetilde{\mathbf{P}} \leftarrow \textsc{DaCAppBas}(\hat{d}, \mathcal{L}^{\mathrm{ab}}_{\hat{\mathbf{D}},\delta}(\hat{\mathbf{F}})\mathbf{X}^{\boldsymbol{\Delta}}, \mathbf{d})$
   $\mathbf{R}^{(2)} \leftarrow$ leading principal $\widetilde{m} \times \widetilde{m}$ submatrix of $\widetilde{\mathbf{P}}$
   $\mathbf{R} \leftarrow \mathbf{R}^{(2)}\mathbf{R}^{(1)}$ `//` $-\widetilde{\boldsymbol{\delta}}$`-ordered weak Popov of` $\mathrm{Syz}_{\mathbf{D}}(\widetilde{\mathbf{F}})$
5. Return the submatrix of $\mathrm{lm}_{-\widetilde{\boldsymbol{\delta}}}(\mathbf{R})^{-1}\mathbf{R}\mathcal{E}$ formed by its rows at indices $\alpha_1 + \cdots + \alpha_i$ for $1 \leqslant i \leqslant m$

Proposition 7.9. *Algorithm 13 is correct and uses*

$$\mathcal{O}(\mathsf{MM}(m, D/m)\log(D/m)) \subseteq \mathcal{O}(m^{\omega}\mathsf{M}(D/m)\log(D/m))$$

operations in $\mathbb{K}$, where $D = D_1 + \cdots + D_n$ and we assume that $D > m$.

Proof. We claim that Steps **2** to **4** correctly compute a $-\widetilde{\boldsymbol{\delta}}$-minimal approximant basis $\mathbf{R}$ of $\mathrm{Syz}_{\mathbf{D}}(\widetilde{\mathbf{F}})$. Then, from Lemma 6.2, we know that the $\mathbf{s}$-Popov approximant basis of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$ is the submatrix of $\mathrm{lm}_{-\widetilde{\boldsymbol{\delta}}}(\mathbf{R})^{-1}\mathbf{R}\mathcal{E}$ formed by its rows at indices $\alpha_1 + \cdots + \alpha_i$ for $1 \leqslant i \leqslant m$, hence the correctness of the algorithm.

To prove our claim, it is enough to show that $\mathbf{R}^{(2)}$ is an $-\hat{\boldsymbol{\delta}}$-minimal approximant basis of $\mathrm{Syz}_{\hat{\mathbf{D}}}(\hat{\mathbf{F}})$. Indeed, according to Proposition 7.6, this implies that $\mathbf{R} = \mathbf{R}^{(2)}\mathbf{R}^{(1)}$ is an approximant basis of $\mathrm{Syz}_{\mathbf{D}}(\widetilde{\mathbf{F}})$; furthermore, since $-\hat{\boldsymbol{\delta}} = \mathrm{rdeg}_{-\widetilde{\boldsymbol{\delta}}}(\mathbf{R}^{(1)})$ and $\mathbf{R}^{(1)}$ is $-\widetilde{\boldsymbol{\delta}}$-reduced, this basis $\mathbf{R}$ is $-\widetilde{\boldsymbol{\delta}}$-reduced according to the second item of Theorem 1.28.

Then, the fact that $\mathbf{R}^{(2)}$ is an $-\hat{\boldsymbol{\delta}}$-minimal approximant basis of $\mathrm{Syz}_{\hat{\mathbf{D}}}(\hat{\mathbf{F}})$ follows from Lemma 7.8, and from the facts that $\hat{\boldsymbol{\delta}}$ is the $-\hat{\boldsymbol{\delta}}$-minimal degree of $\mathrm{Syz}_{\hat{\mathbf{D}}}(\hat{\mathbf{F}})$ and $\delta = \lceil D/m \rceil \geqslant \max(\hat{\boldsymbol{\delta}})$. Indeed, from Lemma 6.2 we know that the $-\widetilde{\boldsymbol{\delta}}$-minimal degree of $\mathrm{Syz}_{\mathbf{D}}(\widetilde{\mathbf{F}})$ is $\widetilde{\boldsymbol{\delta}}$; and if we denote by $\boldsymbol{\delta}^{(1)}$ and $\boldsymbol{\delta}^{(2)}$ the $-\widetilde{\boldsymbol{\delta}}$- and $-\hat{\boldsymbol{\delta}}$-pivot degrees of $\mathbf{R}^{(1)}$ and $\mathbf{R}^{(2)}$, we have $\widetilde{\boldsymbol{\delta}} = \boldsymbol{\delta}^{(1)} + \boldsymbol{\delta}^{(2)}$ according to the fourth item of Theorem 1.28. On the other hand, $-\hat{\boldsymbol{\delta}} = \mathrm{rdeg}_{-\widetilde{\boldsymbol{\delta}}}(\mathbf{R}^{(1)}) = \boldsymbol{\delta}^{(1)} - \widetilde{\boldsymbol{\delta}}$ yields $\widetilde{\boldsymbol{\delta}} = \hat{\boldsymbol{\delta}} + \boldsymbol{\delta}^{(1)}$ and thus $\boldsymbol{\delta}^{(2)} = \hat{\boldsymbol{\delta}}$. In addition, this implies that $\hat{\boldsymbol{\delta}} \leqslant \widetilde{\boldsymbol{\delta}}$ and therefore $\max(\hat{\boldsymbol{\delta}}) \leqslant \max(\widetilde{\boldsymbol{\delta}})$, the latter being at most $\delta = \lceil D/m \rceil$ by construction (see Eq. (6.4)).

Remark also that at Step **4**, Algorithm 11 returns a basis in shifted ordered weak Popov form according to Proposition 7.4; besides, this algorithm only accepts identical orders, hence the use of $\hat{d} = \max(\hat{\mathbf{D}})$ and the corresponding scaling of $\mathcal{L}^{\mathrm{ab}}_{\hat{\mathbf{D}},\delta}(\hat{\mathbf{F}})$ by powers of X.

Concerning the cost bound, Steps **1** and **3** do not involve field operations. Proposition 7.6 indicates that Step **2** uses $\mathcal{O}(\mathsf{MM}(\widetilde{m}, D/\widetilde{m}) \log(D/\widetilde{m}))$ operations, which is within the announced bound since $m \leqslant \widetilde{m} < 2m$ according to Lemma 6.2.

The expanded matrix $\mathcal{L}^{\mathrm{ab}}_{\hat{\mathbf{D}},\delta}(\hat{\mathbf{F}})$ has $\widetilde{m} + \widetilde{\nu}$ rows and $\nu + \widetilde{\nu} < \widetilde{m} + \widetilde{\nu}$ columns, with $\widetilde{\nu} < |\hat{\mathbf{D}}|/\delta$. From Proposition 7.6, we have $|\hat{\mathbf{D}}| \leqslant D$, hence $\widetilde{\nu} < D/\lceil D/m \rceil < m$. As a consequence, $\widetilde{m} + \widetilde{\nu} < 3m$. Furthermore, the orders in $\mathcal{L}^{\mathrm{ab}}_{\hat{\mathbf{D}},\delta}(\hat{\mathbf{D}})$ are all between δ and 2δ by definition, hence $\hat{d} \leqslant 2\lceil D/m \rceil$ is in $\Theta(D/m)$. (Note that we are allowed to discard the ceiling because we have assumed $D > m$). Then, according to Proposition 7.4, the computation of $\widetilde{\mathbf{P}}$ at Step **4** is within the announced bound.

Now, by Proposition 7.6 the degree of $\mathbf{R}^{(1)}$ is at most $2D/\widetilde{m} \leqslant 2D/m$. Besides, $\mathbf{R}^{(2)}$ has $-\hat{\boldsymbol{\delta}}$-pivot degree $\hat{\boldsymbol{\delta}}$ as mentioned above, and thus the column degree of $\mathbf{R}^{(2)}$ is precisely $\hat{\boldsymbol{\delta}}$; this implies that $\deg(\mathbf{R}^{(2)}) \leqslant \max(\hat{\boldsymbol{\delta}}) \leqslant \delta \leqslant 1 + D/m$. Thus the computation of $\mathbf{R} = \mathbf{R}^{(2)}\mathbf{R}^{(1)}$ at Step **4** can be done in $\mathcal{O}(\mathsf{MM}(m, D/m))$ operations.

Finally, the computation of $\mathrm{lm}_{-\widetilde{\boldsymbol{\delta}}}(\mathbf{R})^{-1}$ at Step **5** uses $\mathcal{O}(\widetilde{m}^{\omega})$ operations. Furthermore, since $\mathrm{cdeg}(\mathbf{R}) = \widetilde{\boldsymbol{\delta}}$ and $|\widetilde{\boldsymbol{\delta}}| = |\boldsymbol{\delta}| \leqslant D$ by Lemma 2.10, we can partially linearize the columns of $\mathbf{R}$ to obtain a matrix of row dimension $\widetilde{m}$, column dimension in $\mathcal{O}(\widetilde{m})$, and degree in $\mathcal{O}(D/\widetilde{m})$; then we left-multiply this expanded matrix by $\mathrm{lm}_{-\widetilde{\boldsymbol{\delta}}}(\mathbf{R})^{-1}$ and compress back the columns. Having $\widetilde{m} \in \Theta(m)$, and since right-multiplying by $\mathcal{E}$ does not involve field operations, Step **5** is done in $\mathcal{O}(\mathsf{MM}(m, D/m))$ field operations. ∎

7.4 Fast approximant bases in Popov form for arbitrary shifts

Here, we give a fast algorithm for solving the general case of Problem 7: it supports arbitrary dimensions m and n, orders $\mathbf{D} \in \mathbb{Z}^n_{>0}$, and shift $\mathbf{s} \in \mathbb{Z}^m$, and it returns the shifted Popov approximant basis. Our approach is to rely on the algorithm of the previous section, thanks to a recursive procedure to find the shifted minimal degree of the module of approximants. This allows us to achieve our target cost $\tilde{\mathcal{O}}(m^{\omega-1}D)$ with D being the sum of the orders in $\mathbf{D}$.

Proposition 7.10. *Algorithm 14 is correct and uses*

$$\begin{aligned}&\mathcal{O}(\mathsf{MM}(m, D/m)\log(D/m)^2 + m^{\omega-1}D\log(m))\\ &\subseteq\ \mathcal{O}(m^{\omega}\mathsf{M}(D/m)\log(D/m)^2 + m^{\omega-1}D\log(m)).\end{aligned}$$

operations in $\mathbb{K}$, where $D = D_1 + \cdots + D_n$.

Proof. Concerning the base case of the recursion at Step **1**, Theorem 2.12 shows that it correctly computes the $\mathbf{s}$-Popov approximant basis of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$ using $\mathcal{O}(m^{\omega}\log(m))$ field operations (for the choice of $\mathbf{E}$ and $\mathbf{Z}$, see Section 2.3). When $D > m$, this base case is done less than $2D/m$ times in the overall computation, thus leading to a contribution of $\mathcal{O}(m^{\omega-1}D\log(m))$ in the cost bound.

Let us now study Step **3**, where $D > m$ and $n < m$. The instance $(\mathbf{D}, \mathbf{F})$ is first split into two instances $(\mathbf{D}^{(1)}, \mathbf{F}^{(1)})$ and $(\mathbf{D}^{(2)}, \mathbf{F}^{(2)})$ such that $D^{(1)} = |\mathbf{D}^{(1)}| = \lfloor D/2 \rfloor$ and $D^{(2)} = |\mathbf{D}^{(2)}| = D - |\mathbf{D}^{(1)}| = \lceil D/2 \rceil$. We have $\mathrm{cdeg}(\mathbf{F}^{(1)}) < \mathbf{D}^{(1)}$ and $\mathrm{cdeg}(\mathbf{F}^{(2)}) < \mathbf{D}^{(2)}$. Furthermore, since $n < m$, the column dimension is less than the row dimension for both $\mathbf{F}^{(1)}$ and $\mathbf{F}^{(2)}$, so that the recursive calls at Steps **3.e** and **3.h** will never lead to performing Step **2**. We note that when $d = D_{i_0}$ the first entry of $\mathbf{D}^{(2)}$ is zero; then, one can discard this entry and the corresponding zero column of $\mathbf{F}^{(2)}$.

At Step **3.g**, the residual $\mathbf{G}$ can be computed in $\mathcal{O}(\mathsf{MM}(m, D/m))$ field operations, by a minor variation of Lemma 6.4. Let us recall that, according to Lemma 2.10, the sum of the column degrees of $\mathbf{P}^{(1)}$ is at most $D^{(1)} = \lfloor D/2 \rfloor$. On the other hand, the sum of the entries of $\mathbf{D}^{(2)}$ is $D^{(2)} = \lceil D/2 \rceil$. Our claimed cost for computing $\mathbf{G}$, slightly different from that in Lemma 2.10, then follows by the remark that in the proof of this lemma all operations that are polynomial divisions with remainder are done for free here since the moduli are powers of X.

Let us denote by $\mathbf{t} \in \mathbb{Z}^m$ the shift $\mathbf{t} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}^{(1)}) = \mathbf{s} + \boldsymbol{\delta}^{(1)}$. Assuming that the two recursive calls correctly compute the $\mathbf{s}$- and $\mathbf{t}$-Popov approximant bases $\mathbf{P}^{(1)}$ and $\mathbf{P}^{(2)}$ of $\mathrm{Syz}_{\mathbf{D}^{(1)}}(\mathbf{F}^{(1)})$ and $\mathrm{Syz}_{\mathbf{D}^{(2)}}(\mathbf{F}^{(2)})$, then the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$ is $\boldsymbol{\delta}^{(1)} + \boldsymbol{\delta}^{(2)}$ according to the item (iv) of Theorem 1.28. Then, by Proposition 7.9, Step **3.j** computes the sought approximant basis in $\mathcal{O}(\mathsf{MM}(m, D/m)\log(D/m))$ field operations.

Since the two recursive calls at Steps **3.e** and **3.h** are with the same dimension m and half the total order $D/2$, the announced cost bound follows using the super-linearity property $\mathcal{H}_{\mathsf{MM}(\cdot,\cdot,)}$ of Eq. (6.2).

Finally, Step **2** deals with the case $n \geqslant m$. According to the above discussion, this if statement can only be entered once at the initial call to the algorithm; after that, we

Algorithm 14 – FastPopovAppBas
(Shifted Popov approximant basis)

Input:

- positive integers $\mathbf{D} = (D_1, \ldots, D_n) \in \mathbb{Z}^n_{>0}$,
- matrix $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ with $\operatorname{cdeg}(\mathbf{F}) < \mathbf{D}$,
- shift $\mathbf{s} \in \mathbb{Z}^m$.

Output: the $\mathbf{s}$-Popov approximant basis of $\operatorname{Syz}_{\mathbf{D}}(\mathbf{F})$.

1. If $D = D_1 + \cdots + D_n \leqslant m$ then: `// Base case`

a. For i from 1 to n:

(i) write $\mathbf{F}_{*,i} = \mathbf{f}_i^{(0)} + \mathbf{f}_i^{(1)} X + \cdots + \mathbf{f}_i^{(D_i-1)} X^{D_i-1}$

(ii) $\mathbf{E}_i \leftarrow \begin{bmatrix} \mathbf{f}_i^{(0)} & \mathbf{f}_i^{(1)} & \cdots & \mathbf{f}_i^{(D_i-1)} \end{bmatrix} \in \mathbb{K}^{m \times D_i}$

(iii) $\mathbf{Z}_i \leftarrow \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & 0 & 1 \\ & & & 0 \end{bmatrix} \in \mathbb{K}^{D_i \times D_i}$

b. $\mathbf{E} \leftarrow \begin{bmatrix} \mathbf{E}_1 & \cdots & \mathbf{E}_n \end{bmatrix} \in \mathbb{K}^{m \times D}$; $\mathbf{Z} \leftarrow \operatorname{diag}(\mathbf{Z}_1, \ldots, \mathbf{Z}_n) \in \mathbb{K}^{D \times D}$

c. Return $\textsc{LinPopovRelBas}(\mathbf{E}, \mathbf{Z}, \mathbf{s}, \max(\mathbf{D}))$

2. Else if $n \geqslant m$: `// Entered at most once at initial call`

a. $(\mathbf{P}^{(1)}, \hat{\mathbf{D}}, \hat{\mathbf{F}}, \hat{\mathbf{s}}) \leftarrow \textsc{ReduceNbEqAppBas}(\mathbf{D}, \mathbf{F}, \mathbf{s})$

b. $\mathbf{P}^{(2)} \leftarrow \textsc{FastPopovAppBas}(\hat{\mathbf{D}}, \hat{\mathbf{F}}, \hat{\mathbf{s}})$

c. $\boldsymbol{\delta}^{(1)} \leftarrow$ diagonal degrees of $\mathbf{P}^{(1)}$; $\boldsymbol{\delta}^{(2)} \leftarrow$ diagonal degrees of $\mathbf{P}^{(2)}$

d. Return $\textsc{MinDegAppBas}(\mathbf{D}, \mathbf{F}, \mathbf{s}, \boldsymbol{\delta}^{(1)} + \boldsymbol{\delta}^{(2)})$

3. Else: `// Divide and conquer`

a. $1 \leqslant i_0 \leqslant n$ and $1 \leqslant d \leqslant D_{i_0}$ such that $D_1 + \cdots + D_{i_0-1} + d = \lfloor D/2 \rfloor$

b. $\mathbf{f}^{(1)} = \mathbf{F}_{*,i_0} \bmod X^d$; $\mathbf{f}^{(2)} = X^{-d}(\mathbf{F}_{*,i_0} - \mathbf{f}^{(1)})$

c. $\mathbf{D}^{(1)} \leftarrow (D_1, \ldots, D_{i_0-1}, d)$; $\mathbf{F}^{(1)} \leftarrow [\mathbf{F}_{*,1} | \cdots | \mathbf{F}_{*,i_0-1} | \mathbf{f}^{(1)}]$

d. $\mathbf{D}^{(2)} \leftarrow (D_{i_0} - d, D_{i_0+1}, \ldots, D_n)$; $\mathbf{F}^{(2)} \leftarrow [\mathbf{f}^{(2)} | \mathbf{F}_{*,i_0+1} | \cdots | \mathbf{F}_{*,n}]$

e. $\mathbf{P}^{(1)} \leftarrow \textsc{FastPopovAppBas}(\mathbf{D}^{(1)}, \mathbf{F}^{(1)}, \mathbf{s})$

f. $\boldsymbol{\delta}^{(1)} \leftarrow$ diagonal degrees of $\mathbf{P}^{(1)}$

g. $\mathbf{G} \leftarrow \mathbf{P}^{(1)} \mathbf{F}^{(2)} \bmod \mathbf{X}^{\mathbf{D}^{(2)}}$

h. $\mathbf{P}^{(2)} \leftarrow \textsc{FastPopovAppBas}(\mathbf{D}^{(2)}, \mathbf{G}, \mathbf{s} + \boldsymbol{\delta}^{(1)})$

i. $\boldsymbol{\delta}^{(2)} \leftarrow$ diagonal degrees of $\mathbf{P}^{(2)}$

j. Return $\textsc{MinDegAppBas}(\mathbf{D}, \mathbf{F}, \mathbf{s}, \boldsymbol{\delta}^{(1)} + \boldsymbol{\delta}^{(2)})$

always have $n < m$ in the recursive calls induced by Step **2.b**. If it is the case that $n \geqslant m$ at the initial call, this step relies on Algorithm 12 to reduce the number of equations and then deal with the remaining equations recursively. The correctness and cost bound then follow from Proposition 7.6 and the same arguments as those used above concerning the correctness and cost bound of Step **3**. ☙

8

Computing shifted Popov solution bases

In this chapter, we give a fast algorithm to compute solution bases as in Problem 9. Under the assumption that the number of equations is not much larger than the number of unknowns, our algorithm computes the shifted Popov basis for arbitrary shifts and arbitrary moduli with the same cost bound as for approximant bases (Algorithm 14), up to logarithmic factors. We recall that in the specific case of approximants, all moduli are powers of the variable.

In Section 8.1, we first give details of a link between solution bases and kernel bases in shifted Popov forms. Using the fast approximant basis algorithm developed in Chapter 7, we show how to compute such shifted Popov kernel bases efficiently. Then, combining this with partial linearization to ensure that the degrees in the ouput basis are uniformly small, this gives us a fast algorithm to compute solution bases when the shifted minimal degree is known a priori.

To find this minimal degree, we then follow a divide-and-conquer approach similar to that in Algorithm 14, presented in Section 8.3. However, here we have no access to factors of the moduli, which are arbitrary polynomials given by their coefficients; as a result, we will only split the instance according to the number of such moduli, that is, the number of equations.

At the base case of this recursion, we are thus faced with the case of one equation. This question is studied in Section 8.2, where we propose a new efficient strategy based on the following ideas. As mentioned above, we may rewrite the problem as a kernel basis problem; here, the matrix in input is a column vector. When the shift in input has small entries, such a kernel basis can be computed with efficiently via a single call to a fast approximant basis algorithm. On the other hand, we remark that when the entries of the shift have a large amplitude, the output must have some block triangular shape which, if known, can be used to divide the problem into smaller subproblems. We then design a recursive approach which splits the shift into two subshifts of half the amplitude, rely on a first recursive call to reveal a part of this block triangular shape, then use this shape to define the second subproblem which we solve by a second recursive call, and finally gather the results by deducing the shifted minimal degree and computing the solution basis from the known minimal degree.

8.1 Fast algorithm via kernel bases when the minimal degree is known

This section summarizes and extends results from [GS11, Section 3]. We first show that the $\mathbf{s}$-Popov solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ is the principal $m \times m$ submatrix of the $\mathbf{u}$-Popov kernel basis of $[\mathbf{F}^\mathsf{T} \;\; \mathrm{diag}(\mathfrak{M})]^\mathsf{T}$ for some shift $\mathbf{u} \in \mathbb{Z}^{m+n}$. While it is not known how to perform such kernel computations efficiently in general, this link will still lead us to a fast algorithm for computing solution bases when the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ is known.

We start by recalling the notion of kernel bases for polynomial matrices.

Definition 8.1 (Kernel)**.** *Let $\mathbf{V} \in \mathbb{K}[X]^{m\times n}$ be some polynomial matrix of rank ρ. Then, the* kernel of $\mathbf{V}$ *is the $\mathbb{K}[X]$-module*

$$\{\mathbf{p} \in \mathbb{K}[X]^{1\times m} \mid \mathbf{pV} = 0\}$$

of rank $m - \rho$, and a kernel basis for $\mathbf{V}$ *is a matrix $\mathbf{N} \in \mathbb{K}[X]^{\rho\times m}$ whose rows form a basis of the kernel of $\mathbf{V}$.*

Then, we have the following relationship between solution bases and kernel bases.

Lemma 8.2. *Let $\mathfrak{M} = (\mathfrak{m}_1, \ldots, \mathfrak{m}_n) \in \mathbb{K}[X]^n_{\neq 0}$, let $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ be such that $\deg(\mathbf{F}_{*,j}) < \deg(\mathfrak{m}_j)$ for $1 \leqslant j \leqslant n$, let $\mathbf{s} \in \mathbb{Z}^m$. Let further $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$, let $\mathbf{w} \in \mathbb{Z}^n$ be any shift such that $\max(\mathbf{w}) \leqslant \min(\mathbf{s})$, and let $\mathbf{u} = (\mathbf{s}, \mathbf{w}) \in \mathbb{Z}^{m+n}$.*

Then, $\mathbf{P}$ is the $\mathbf{s}$-Popov solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ if and only if

$$\begin{bmatrix}\mathbf{P} & \mathbf{Q}\end{bmatrix} \text{ is the } \mathbf{u}\text{-Popov kernel basis for } \begin{bmatrix}\mathbf{F} \\ \mathrm{diag}(\mathfrak{M})\end{bmatrix} \text{ for some } \mathbf{Q} \in \mathbb{K}[X]^{m\times n};$$

in this case, $\deg(\mathbf{Q}) < \deg(\mathbf{P})$ and $[\mathbf{P} \;\; \mathbf{Q}]$ has $\mathbf{s}$-pivot index $(1, 2, \ldots, m)$.

Proof. Let $\mathbf{D} = \mathrm{diag}(\mathfrak{M})$, and let $\mathbf{V} = [\mathbf{F}^\mathsf{T} \;\; \mathbf{D}]^\mathsf{T} \in \mathbb{K}[X]^{(m+n)\times n}$. We first verify that $\mathbf{P}$ is a solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ if and only if there is some $\mathbf{Q} \in \mathbb{K}[X]^{m\times n}$ such that $[\mathbf{P} \;\; \mathbf{Q}]$ is a kernel basis for $\mathbf{V}$.

First assume that $\mathbf{P}$ is a solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$. Then, $\mathbf{Q} = -\mathbf{PFD}^{-1} \in \mathbb{K}[X]^{m\times n}$ is such that $[\mathbf{P} \;\; \mathbf{Q}]\mathbf{V} = 0$. Let us consider $[\mathbf{p} \;\; \mathbf{q}] \in \mathbb{K}[X]^{1\times(m+n)}$ in the kernel of $\mathbf{V}$ and show that it is a $\mathbb{K}[X]$-linear combination of the rows of $[\mathbf{P} \;\; \mathbf{Q}]$. Indeed, $\mathbf{p}$ is a solution of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$, so that $\mathbf{p} = \boldsymbol{\lambda}\mathbf{P}$ for some $\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m}$. Then, $-\mathbf{qD} = \mathbf{pF} = \boldsymbol{\lambda}\mathbf{PF} = -\boldsymbol{\lambda}\mathbf{QD}$ yields $\mathbf{q} = \boldsymbol{\lambda}\mathbf{Q}$, hence $[\mathbf{p} \;\; \mathbf{q}] = \boldsymbol{\lambda}[\mathbf{P} \;\; \mathbf{Q}]$.

Now, assume that there exists $\mathbf{Q} \in \mathbb{K}[X]^{m\times n}$ such that $[\mathbf{P} \;\; \mathbf{Q}]$ is a kernel basis for $\mathbf{V}$. Then, in particular, $\mathbf{PF} = 0 \bmod \mathfrak{M}$. We consider a solution $\mathbf{p}$ of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ and show that it is a $\mathbb{K}[X]$-linear combination of the rows of $\mathbf{P}$. Defining $\mathbf{q} = -\mathbf{pFD}^{-1}$, we have $[\mathbf{p} \;\; \mathbf{q}]\mathbf{V} = 0$, hence $[\mathbf{p} \;\; \mathbf{q}] = \boldsymbol{\lambda}[\mathbf{P} \;\; \mathbf{Q}]$ for some $\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m}$, and therefore $\mathbf{p} = \boldsymbol{\lambda}\mathbf{P}$.

Finally, having $\deg(\mathbf{F}_{*,j}) < \deg(\mathfrak{m}_j)$ implies that any $[\mathbf{p} \;\; \mathbf{q}] \in \mathbb{K}[X]^{1\times(m+n)}$ in the kernel of $\mathbf{V}$ satisfies $\deg(\mathbf{q}) < \deg(\mathbf{p})$, and thus from $\max(\mathbf{w}) \leqslant \min(\mathbf{s})$ we obtain that $\mathrm{rdeg}_{\mathbf{w}}(\mathbf{q}) < \mathrm{rdeg}_{\mathbf{s}}(\mathbf{p})$. In particular, for any matrix $[\mathbf{P} \;\; \mathbf{Q}] \in \mathbb{K}[X]^{m\times(m+n)}$ such that $[\mathbf{P} \;\; \mathbf{Q}]\mathbf{V} = 0$, we have $\mathrm{lm}_{\mathbf{u}}([\mathbf{P} \;\; \mathbf{Q}]) = [\mathrm{lm}_{\mathbf{s}}(\mathbf{P}) \;\; \mathbf{0}]$. This implies that $\mathbf{P}$ is in $\mathbf{s}$-Popov form if and only if $[\mathbf{P} \;\; \mathbf{Q}]$ is in $\mathbf{u}$-Popov form with $\mathbf{s}$-pivot index $(1, \ldots, m)$. ∎

While this provides an algorithm to compute solution bases via kernel bases, to the best of our knowledge, there is currently no known algorithm which would compute kernel bases in shifted Popov form for arbitrary shifts within a number of operations that matches our target cost bound $\tilde{\mathcal{O}}(m^{\omega-1}D)$, where $D = \deg(\mathfrak{m}_1) + \cdots + \deg(\mathfrak{m}_n)$.

Even if we had such an algorithm, an obstacle of this approach is that it increases the matrix dimension of the problem: instead of considering $\mathbf{F}$ with m rows, we consider $\mathbf{V}$ with $m + n$ rows. In fact, we will not manage to overcome this issue; this is one of the reasons why we obtain the cost bound $\tilde{\mathcal{O}}(m^{\omega-1}D)$ only under the assumption $n \in \mathcal{O}(m)$. (This is not the only reason: our main algorithm in Section 8.3 computes a residual as in Lemma 6.4, and in this lemma we also required $n \in \mathcal{O}(m)$ for efficiency.)

Another immediate difficulty is that this method does not compute only the solution basis $\mathbf{P}$ but also the matrix $\mathbf{Q} \in \mathbb{K}[X]^{m\times n}$ formed by the quotients $\mathbf{Q} = -\mathbf{PF}/\mathrm{diag}(\mathfrak{M})$. Yet, $\mathbf{Q}$ may have size $\mathcal{O}(mnD)$ when $\mathbf{P}$ has columns of large degree as we show in the next example; this size is beyond our target cost bound $\tilde{\mathcal{O}}(m^{\omega-1}D)$.

Example 8.3. Here, we work over the finite field $\mathbb{K} = \mathbb{F}_{997}$ and with the matrix dimensions $m = n = 4$. With a computer algebra system, we choose at random monic moduli $(\mathfrak{m}_1, \mathfrak{m}_2, \mathfrak{m}_3, \mathfrak{m}_4)$ of respective degrees $(5, 10, 3, 19)$, as well as a matrix $\mathbf{F} \in \mathbb{K}[X]^{4\times 4}$ of column degree $\mathrm{cdeg}(\mathbf{F}) < (5, 10, 3, 19)$. We thus have $D = 5 + 10 + 3 + 19 = 37$.

Then, we define the matrix $\mathbf{V} \in \mathbb{K}[X]^{8\times 4}$ as

$$\mathbf{V} = \begin{bmatrix} \mathbf{F} \\ \mathrm{diag}(\mathfrak{M}) \end{bmatrix} = \begin{bmatrix} [4] & [9] & [2] & [18] \\ [4] & [9] & [2] & [18] \\ [4] & [9] & [2] & [18] \\ [4] & [9] & [2] & [18] \\ (5) & & & \\ & (10) & & \\ & & (3) & \\ & & & (19) \end{bmatrix},$$

where $[d]$ indicates an entry of degree at most d, (d) indicates a monic entry of degree d, and blank entries denote the zero polynomial.

Now, we compute the $(0, 37, 74, 111, 0, 0, 0, 0)$-Popov kernel basis for $\mathbf{V}$, obtaining the matrix $[\mathbf{P} \;\; \mathbf{Q}] \in \mathbb{K}[X]^{4\times 8}$, whose degrees are

$$\begin{bmatrix} \mathbf{P} & \mathbf{Q} \end{bmatrix} = \begin{bmatrix} (37) & & & & [36] & [36] & [36] & [36] \\ [36] & (0) & & & [35] & [35] & [35] & [35] \\ [36] & & (0) & & [35] & [35] & [35] & [35] \\ [36] & & & (0) & [35] & [35] & [35] & [35] \end{bmatrix}.$$

In this case, we note that $\mathbf{P}$ is the Hermite solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$. Furthermore, the sum of column degrees of $\mathbf{P}$ is $|\mathrm{cdeg}(\mathbf{P})| = \deg(\det(\mathbf{P})) = 37 = D$, as one could expect from Lemma 2.10. However, $\mathbf{P}$ has a column of large degree; by large, we mean compared to the average $37/4$. As a consequence, the quotients in $\mathbf{Q}$ all have degree close to D.

Generalizing this example to arbitrary dimensions $m = n$ and D, we get instances where the quotient matrix $\mathbf{Q}$ has size in $\Theta(m^2 D)$, and thus cannot be computed in $\mathcal{O}(m^{\omega-1}D)$ field operations. ✍

Still, these quotients are not part of our specification of Problem 9: they are not wanted in our context. Thanks to this remark, there is a specific situation in which we are able to circumvent this difficulty concerning the size of $\mathbf{Q}$, and to take advantage of this interpretation of solution bases as submatrices of kernel bases.

Namely, let us now place ourselves in the case where the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ is known a priori. Then, partial linearization techniques can be used to reduce to the case of almost uniform column degrees in the output. This has been presented for relation bases in general in Section 6.2, and we give below a slightly modified version of Lemma 6.2 specialized to the particular case of solution bases.

Corollary 8.4. *Let $\mathfrak{M} \in \mathbb{K}[X]^n_{\neq 0}$ with its entries having degrees $(D_1, \ldots, D_n)$, let $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ be such that $\mathrm{cdeg}(\mathbf{F}) < (D_1, \ldots, D_n)$, and let $\mathbf{s} \in \mathbb{Z}^m$. Furthermore, let $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m)$ denote the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$.*

Let $\delta = \lceil (D_1 + \cdots + D_n)/m \rceil \geqslant 1$, and for $i \in \{1, \ldots, m\}$ write $\delta_i = (\alpha_i - 1)\delta + \beta_i$ with $\alpha_i = \lceil \delta_i/\delta \rceil$ and $1 \leqslant \beta_i \leqslant \delta$ if $\delta_i > 0$, and with $\alpha_i = 1$ and $\beta_i = 0$ if $\delta_i = 0$. Then, let let $\widetilde{m} = \alpha_1 + \cdots + \alpha_m$, and define $\widetilde{\boldsymbol{\delta}} \in \mathbb{Z}^{\widetilde{m}}_{\geqslant 0}$ as in Eq. (6.4),

$$\widetilde{\boldsymbol{\delta}} = (\underbrace{\delta, \ldots, \delta, \beta_1}_{\alpha_1}, \ldots, \underbrace{\delta, \ldots, \delta, \beta_m}_{\alpha_m})$$

and the expansion-compression matrix $\mathcal{E} \in \mathbb{K}[X]^{\widetilde{m}\times m}$ as in Eq. (6.5),

$$\mathcal{E} = \begin{bmatrix} 1 & & \\ X^{\delta} & & \\ \vdots & & \\ X^{(\alpha_1-1)\delta} & & \\ & \ddots & \\ & & 1 \\ & & X^{\delta} \\ & & \vdots \\ & & X^{(\alpha_m-1)\delta} \end{bmatrix}.$$

Let $\mathbf{d} = -\widetilde{\boldsymbol{\delta}} \in \mathbb{Z}^{\widetilde{m}}$ and $\widetilde{\mathbf{P}} \in \mathbb{K}[X]^{\widetilde{m}\times\widetilde{m}}$ be the $\mathbf{d}$-Popov solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathcal{E}\mathbf{F} \bmod \mathfrak{M})$. Then, $\widetilde{\mathbf{P}}$ has $\mathbf{d}$-pivot degree $\widetilde{\boldsymbol{\delta}}$ and the $\mathbf{s}$-Popov solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ is the submatrix of $\widetilde{\mathbf{P}}\mathcal{E}$ formed by its rows at indices $\{\alpha_1 + \cdots + \alpha_i, 1 \leqslant i \leqslant m\}$. Furthermore, we have $m \leqslant \widetilde{m} \leqslant 2m$ and $\max(\mathbf{d}) - \min(\mathbf{d}) \leqslant \delta$.

In terms of algorithm, this result leads us to perform a transformation of the input and to compute a solution basis which is a partially linearized version of the sought one. Computing this linearized basis via a kernel basis, we will at the same time compute a corresponding quotient matrix which, unlike the solution basis, does not directly yield the actual quotient matrix. This is not a problem since our aim is only to compute the solution basis.

Thus, to compute solution bases with known minimal degree, we will resort to an efficient procedure to compute shifted Popov kernel bases when information on the pivot

entries of the output is available. The next result indicates that this can be done via an approximant basis computation.

Lemma 8.5. *Let* $\mathbf{V} \in \mathbb{K}[X]^{(m+n)\times n}$ *have full rank and* $\mathbf{s} \in \mathbb{Z}^{m+n}$. *Let* $\mathbf{N} \in \mathbb{K}[X]^{m\times(m+n)}$ *be the* $\mathbf{s}$*-Popov kernel basis for* $\mathbf{V}$, *let* $(\pi_1, \ldots, \pi_m)$ *be its* $\mathbf{s}$*-pivot index,* $(\delta_1, \ldots, \delta_m)$ *be its* $\mathbf{s}$*-pivot degree, and* $\beta \geqslant \deg(\mathbf{N})$ *be a degree bound. Let further* $\mathbf{v} = (v_1, \ldots, v_{m+n}) \in \mathbb{Z}^{m+n}_{\leqslant 0}$ *be defined by*

$$v_j = \begin{cases} -\beta - 1 & \text{if } j \notin \{\pi_1, \ldots, \pi_m\}, \\ -\delta_i & \text{if } j = \pi_i. \end{cases}$$

Then, let $\boldsymbol{\tau} = \operatorname{cdeg}(\mathbf{V}) + (\beta+1, \ldots, \beta+1) \in \mathbb{Z}^n_{>0}$, *and* $\mathbf{A}$ *be the* $\mathbf{v}$*-Popov approximant basis of* $\operatorname{Syz}_{\boldsymbol{\tau}}(\mathbf{V})$. *Then,* $\mathbf{N}$ *is the submatrix of* $\mathbf{A}$ *formed by its rows at indices* $\{\pi_1, \ldots, \pi_m\}$.

Proof. First, $\mathbf{N}$ is in $\mathbf{v}$-Popov form with $\operatorname{rdeg}_{\mathbf{v}}(\mathbf{N}) = \mathbf{0}$. Define $\mathbf{B} \in \mathbb{K}[X]^{(m+n)\times(m+n)}$ whose i-th row is $\mathbf{N}_{j,*}$ if $i = \pi_j$ and $\mathbf{A}_{i,*}$ if $i \notin \{\pi_1, \ldots, \pi_m\}$: we want to prove $\mathbf{B} = \mathbf{A}$.

Writing $(d_1, \ldots, d_n)$ for the column degree of $\mathbf{V}$, we have $\boldsymbol{\tau} = [\tau_j]_j$ with $\tau_j = d_j + \beta + 1$ for $1 \leqslant j \leqslant n$.

Let $\mathbf{p} = [p_j]_j \in \mathbb{K}[X]^{1\times(m+n)}$ be a row of $\mathbf{A}$, and assume $\operatorname{rdeg}_{\mathbf{v}}(\mathbf{p}) < 0$. This means $\deg(p_j) < -v_j$ for all j, so that $\deg(\mathbf{p}) < \max(-\mathbf{v}) = \beta + 1$. Then, we have $\deg(\mathbf{p}\mathbf{V}_{*,j}) < d_j + \beta + 1 = \tau_j$ for all $1 \leqslant j \leqslant n$, and from $\mathbf{p}\mathbf{V}_{*,j} = 0 \bmod X^{\tau_j}$ we obtain $\mathbf{p}\mathbf{V}_{*,j} = 0$, which is absurd by minimality of $\mathbf{N}$. As a result, $\operatorname{rdeg}_{\mathbf{v}}(\mathbf{A}) \geqslant \mathbf{0} = \operatorname{rdeg}_{\mathbf{v}}(\mathbf{N})$ componentwise.

Besides, $\mathbf{B}\mathbf{F} = 0 \bmod (X^{\tau_1}, \ldots, X^{\tau_n})$ and since $\mathbf{B}$ has its $\mathbf{v}$-pivot entries on the diagonal, it is $\mathbf{v}$-reduced: by minimality of $\mathbf{A}$, we obtain $\operatorname{rdeg}_{\mathbf{v}}(\mathbf{A}) = \operatorname{rdeg}_{\mathbf{v}}(\mathbf{B})$. Then, it is easily verified that $\mathbf{B}$ is in $\mathbf{v}$-Popov form, hence $\mathbf{B} = \mathbf{A}$. ☙

In particular, using Algorithm 14, one can efficiently compute the shifted Popov kernel basis for $\mathbf{V}$ if a degree bound β, the shifted pivot index, and the shifted pivot degree are known a priori.

Proposition 8.6. *Algorithm 15 is correct and uses* $\mathcal{O}\tilde{}(m^{\omega-1}(d + n\beta))$ *operations in* $\mathbb{K}$, *where* $d = |\operatorname{cdeg}(\mathbf{V})|$ *is the sum of the column degrees of* $\mathbf{V}$.

Proof. The correctness of Algorithm 15 follows from Lemma 8.5. Besides, the approximant basis computation at Step **3** can be performed in $\mathcal{O}\tilde{}(m^{\omega-1}(D + n\beta))$ according to Proposition 7.10. ☙

In the context of the computation of solution bases with known minimal degree $\boldsymbol{\delta}$, the kernel basis is $[\mathbf{P} \;\; \mathbf{Q}]$ and we have $\deg(\mathbf{Q}) < \deg(\mathbf{P})$ (see Lemma 8.2). Therefore we would choose $\beta = \max(\boldsymbol{\delta}) \geqslant \deg([\mathbf{P} \;\; \mathbf{Q}])$. As noted above, when $\boldsymbol{\delta}$ has some large entries we may have $\beta = \Theta(D)$ and this kernel basis algorithm has cost bound $\mathcal{O}\tilde{}(m^{\omega-1}(D + nD))$, which exceeds our target $\mathcal{O}\tilde{}(m^{\omega-1}D)$. Yet, thanks to Corollary 8.4 we can make $\boldsymbol{\delta}$ close to uniform and thus solve this issue, leading us to Algorithm 16.

Proposition 8.7. *Algorithm 16 is correct and uses* $\mathcal{O}\tilde{}(m^{\omega-1}D)$ *operations in* $\mathbb{K}$, *where* $D = \deg(\mathfrak{m}_1) + \cdots + \deg(\mathfrak{m}_n)$ *and it is assumed that* $D \geqslant m$ *and* $n \in \mathcal{O}(m)$.

Algorithm 15 – PivDegKerBas
(Shifted Popov kernel basis with known pivot degree)

Input:

- a matrix $\mathbf{V} \in \mathbb{K}[X]^{(m+n)\times n}$ with full rank,
- a shift $\mathbf{s} \in \mathbb{Z}^{m+n}$,
- the $\mathbf{s}$-pivot index $(\pi_1, \ldots, \pi_m)$ of the $\mathbf{s}$-Popov kernel basis for $\mathbf{V}$,
- the $\mathbf{s}$-pivot degree $(\delta_1, \ldots, \delta_m)$ of the $\mathbf{s}$-Popov kernel basis for $\mathbf{V}$,
- a bound $\beta \in \mathbb{Z}_{>0}$ at least the degree of the $\mathbf{s}$-Popov kernel basis for $\mathbf{V}$.

Output: the $\mathbf{s}$-Popov kernel basis for $\mathbf{V}$.

1. $\mathbf{v} \leftarrow (v_1, \ldots, v_{m+n}) \in \mathbb{Z}^{m+n}$ with $v_j = -\beta - 1$ if $j \notin \{\pi_1, \ldots, \pi_m\}$ and $v_j = -\delta_i$ if $j = \pi_i$.
2. $\boldsymbol{\tau} \leftarrow \operatorname{cdeg}(\mathbf{V}) + (\beta+1, \ldots, \beta+1) \in \mathbb{Z}^n_{>0}$
3. $\mathbf{A} \leftarrow$ FastPopovAppBas$(\boldsymbol{\tau}, \mathbf{V}, \mathbf{v})$
4. Return the submatrix of $\mathbf{A}$ formed by its rows $\{\pi_1, \ldots, \pi_m\}$

Proof. First, since we have chosen $\mathbf{u} = (-\widetilde{\boldsymbol{\delta}}, -\delta, \ldots, -\delta)$ with $-\delta = \min(-\widetilde{\boldsymbol{\delta}})$, it follows from Lemma 8.2 that the $\mathbf{u}$-Popov kernel basis $\mathbf{N}$ for $[\widetilde{\mathbf{F}}^\mathsf{T} \;\; \operatorname{diag}(\mathfrak{M})]^\mathsf{T}$ can be written as $[\widetilde{\mathbf{P}} \;\; \widetilde{\mathbf{Q}}]$ where $\widetilde{\mathbf{P}}$ is the $-\widetilde{\boldsymbol{\delta}}$-Popov solution basis of $\operatorname{Syz}_{\mathfrak{M}}(\widetilde{\mathbf{F}})$. This lemma also ensures that $\mathbf{N}$ has $\mathbf{u}$-pivot index $\boldsymbol{\pi} = (1, \ldots, m)$ and $\mathbf{u}$-pivot degree $\widetilde{\boldsymbol{\delta}}$; and since $\deg(\widetilde{\mathbf{Q}}) < \deg(\widetilde{\mathbf{P}})$, we have $\deg(\mathbf{N}) \leqslant \deg(\widetilde{\mathbf{P}}) = \max(\widetilde{\boldsymbol{\delta}}) = \delta$.

Then, Proposition 8.6 states that this kernel basis $\mathbf{N}$ is correctly computed at Step **3**, and Corollary 8.4 ensures that the submatrix at Step **4** is the sought $\mathbf{s}$-Popov solution basis of $\operatorname{Syz}_{\mathfrak{M}}(\mathbf{F})$.

Concerning the cost bound, we recall from the proof of Lemma 6.4 that $\mathcal{E}\mathbf{F} \bmod \mathfrak{M}$ at Step **2** can be computed in $\tilde{\mathcal{O}}(mD)$ operations, assuming $n \in \mathcal{O}(m)$. Now, according to Proposition 8.6, the kernel basis computation at Step **3** uses

$$\tilde{\mathcal{O}}(m^{\omega-1}(|\operatorname{cdeg}(\mathbf{V})| + n\delta)) \;\subseteq\; \tilde{\mathcal{O}}(m^{\omega-1}D)$$

operations in $\mathbb{K}$. In this bound, the inclusion follows first from the identity $|\operatorname{cdeg}(\mathbf{V})| = \deg(\mathfrak{m}_1) + \cdots + \deg(\mathfrak{m}_n) = D$, and second from the inequality $n\delta = n\lceil D/m \rceil < n + nD/m$, with $n + nD/m \in \mathcal{O}(D)$ since by assumption $n \in \mathcal{O}(m)$ and $m \leqslant D$. ☙

We remark that when $D \leqslant m$, one may rely on our algorithm based on fast linear algebra (Algorithm 3 in Section 4.3.2) to compute the shifted Popov solution basis in $\tilde{\mathcal{O}}(mD^{\omega-1})$ operations, without requiring that the shifted minimal degree be known.

8.2 The case of one equation

We now present the main new ingredients which allow us to efficiently deal with arbitrary moduli, instead of powers of X as in Chapter 7. These ingredients concern the case of a

Algorithm 16 – MinDegSolBas
(Solution basis with known minimal degree)

Input:

- polynomials $\mathfrak{M} = (\mathfrak{m}_1, \ldots, \mathfrak{m}_n) \in \mathbb{K}[X]^n_{\neq 0}$,
- a matrix $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ with $\deg(\mathbf{F}_{*,j}) < \deg(\mathfrak{m}_j)$,
- a shift $\mathbf{s} \in \mathbb{Z}^m$,
- $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m)$ the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$.

Output: the $\mathbf{s}$-Popov solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$.

1. `/* Partial linearization parameters */`
   $\delta \leftarrow \lceil (\deg(\mathfrak{m}_1) + \cdots + \deg(\mathfrak{m}_n))/m \rceil$,
   $\alpha_i \leftarrow \lfloor \delta_i/\delta \rfloor + 1$ for $1 \leqslant i \leqslant m$,
   $\widetilde{m} \leftarrow \alpha_1 + \cdots + \alpha_m$,
   $\widetilde{\boldsymbol{\delta}} \leftarrow$ tuple as in Eq. (6.4),
   $\mathcal{E} \leftarrow$ matrix as in Eq. (6.5)
2. `/* Partially linearize the equations */`
   $\widetilde{\mathbf{F}} \leftarrow \mathcal{E}\mathbf{F} \bmod \mathfrak{M}$
3. `/* Reduce to kernel basis */`
   $\mathbf{V} \leftarrow [\widetilde{\mathbf{F}}^{\mathsf{T}} \;\; \mathrm{diag}(\mathfrak{M})]^{\mathsf{T}}$
   $\mathbf{u} \leftarrow (-\widetilde{\boldsymbol{\delta}}, -\delta, \ldots, -\delta) \in \mathbb{Z}^{\widetilde{m}+n}$
   $\boldsymbol{\pi} \leftarrow (1, \ldots, m)$
   $\mathbf{N} \leftarrow$ PivDegKerBas$(\mathbf{V}, \mathbf{u}, \boldsymbol{\pi}, \widetilde{\boldsymbol{\delta}}, \delta)$
4. `/* Retrieve the expanded basis and compress back */`
   Write $\mathbf{N} = [\widetilde{\mathbf{P}} \;\; \widetilde{\mathbf{Q}}]$ with $\widetilde{\mathbf{P}} \in \mathbb{K}[X]^{\widetilde{m} \times \widetilde{m}}$
   Return the submatrix of $\widetilde{\mathbf{P}}\mathcal{E}$ formed by its rows $\{\alpha_1 + \cdots + \alpha_i, 1 \leqslant i \leqslant m\}$

single equation, that is, $n = 1$, for which they lead to an efficient algorithm to find the shifted Popov solution basis. Then, we show in Section 8.3 how to rely on this as the base case of a divide-and-conquer scheme on the number of equations.

8.2.1 Amplitude, splitting indices, and block triangular shape

First, we show that when the shift $\mathbf{s}$ has a small *amplitude* $\operatorname{amp}(\mathbf{s}) = \max(\mathbf{s}) - \min(\mathbf{s})$, one can solve Problem 9 via an approximant basis computation at small order.

Lemma 8.8. *Let $\mathfrak{m} \in \mathbb{K}[X]_{\neq 0}$ be of degree D, let $\mathbf{F} \in \mathbb{K}[X]^{m\times 1}$ with $\deg(\mathbf{F}) < D$, and let $\mathbf{s} \in \mathbb{Z}^m$. Consider the shift $\mathbf{u} = (\mathbf{s}, \min(\mathbf{s})) \in \mathbb{Z}^{m+1}$ and the polynomial vector $\mathbf{V} = [\mathbf{F}^\mathsf{T} \;\; \mathfrak{m}]^\mathsf{T} \in \mathbb{K}[X]^{(m+1)\times 1}$. For any $\tau \geqslant \operatorname{amp}(\mathbf{s}) + 2D$, the $\mathbf{s}$-Popov solution basis of $\operatorname{Syz}_{\mathfrak{m}}(\mathbf{F})$ is the principal $m \times m$ submatrix of the $\mathbf{u}$-Popov approximant basis of $\operatorname{Syz}_\tau(\mathbf{V})$.*

Proof. Let us denote by $\mathbf{A} = \left[\begin{smallmatrix} \mathbf{P} & \mathbf{q} \\ \mathbf{p} & q \end{smallmatrix}\right] \in \mathbb{K}[X]^{(m+1)\times(m+1)}$ the $\mathbf{u}$-Popov approximant basis of $\operatorname{Syz}_\tau(\mathbf{V})$, where $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ and $q \in \mathbb{K}[X]$, and let $\mathbf{B} = [\bar{\mathbf{P}} \;\; \bar{\mathbf{q}}] \in \mathbb{K}[X]^{m\times(m+1)}$ be the $\mathbf{u}$-Popov kernel basis for $\mathbf{V}$. Then, by Lemma 8.2, it is enough to prove that $\mathbf{B} = [\mathbf{P} \;\; \mathbf{q}]$.

First, since the smallest entry of $\mathbf{u}$ is its last one, and since $\mathbf{A}$ is in $\mathbf{u}$-Popov form, we have $\operatorname{rdeg}(\mathbf{p}) \leqslant \deg(q)$. This implies that $q\mathfrak{m} \neq 0$, and using $\deg(\mathbf{F}) < D = \deg(\mathfrak{m})$ we obtain that $\deg(\mathbf{pF} + q\mathfrak{m}) = \deg(q) + D$. Since $\mathbf{pF} + q\mathfrak{m} = 0 \bmod X^\tau$, this yields the inequality $\deg(q) + D \geqslant \tau$.

Furthermore, since we have $\mathbf{BV} = 0$, the same arguments also prove that the $\mathbf{u}$-pivot entries of $\mathbf{B}$ are located in $\bar{\mathbf{P}}$.

Now, since the sum of the $\mathbf{u}$-pivot degrees of $\mathbf{A}$ is at most τ by Lemma 2.10, then the sum of the $\mathbf{s}$-pivot degrees of $\mathbf{P}$ is at most D. With $[\mathbf{P} \;\; \mathbf{q}]$ in $\mathbf{u}$-Popov form, this implies that $\deg(\mathbf{q}) < D + \operatorname{amp}(\mathbf{s}) \leqslant \tau - D$. We obtain $\deg(\mathbf{PF} + \mathbf{q}\mathfrak{m}) < \tau$, hence $\mathbf{PF} + \mathbf{q}\mathfrak{m} = 0$. Thus, the minimality of $\mathbf{B}$ and $\mathbf{A}$ gives the conclusion. ∎

When $\operatorname{amp}(\mathbf{s}) \in \mathcal{O}(D)$, this gives a fast solution to our problem. In what follows, we present a divide-and-conquer approach on $\operatorname{amp}(\mathbf{s})$, with base case $\operatorname{amp}(\mathbf{s}) \in \mathcal{O}(D)$.

We first give an overview of this approach, assuming that $\mathbf{s}$ is non-decreasing. A key ingredient is that when $\operatorname{amp}(\mathbf{s})$ is large compared to D, then $\mathbf{P}$ has a lower block triangular shape, since it is in $\mathbf{s}$-Popov form with sum of $\mathbf{s}$-pivot degrees $|\boldsymbol{\delta}| \leqslant D$. Typically, having $s_{i+1} - s_i \geqslant D$ for some i implies that

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}^{(1)} & \mathbf{0} \\ * & \mathbf{P}^{(2)} \end{bmatrix}$$

with $\mathbf{P}^{(1)} \in \mathbb{K}[X]^{i\times i}$ and $\mathbf{P}^{(2)} \in \mathbb{K}[X]^{(m-i)\times(m-i)}$. In general, even though the block sizes are unknown a priori, we are going to show that they can be revealed efficiently, along with the $\mathbf{s}$-minimal degree $\boldsymbol{\delta}$ of $\operatorname{Syz}_{\mathfrak{m}}(\mathbf{F})$, by a divide-and-conquer algorithm that we sketch in the next paragraphs.

First, we use a recursive call with the first j entries $\mathbf{s}^{(0)}$ of $\mathbf{s}$ and $\mathbf{F}^{(0)}$ of $\mathbf{F}$, where j is such that $\operatorname{amp}(\mathbf{s}^{(0)})$ is about half of $\operatorname{amp}(\mathbf{s})$. Computing the $\mathbf{s}$-Popov solution basis of $\operatorname{Syz}_{\mathfrak{m}}(\mathbf{F}^{(0)})$ reveals the first $i \leqslant j$ entries $\boldsymbol{\delta}^{(1)}$ of $\boldsymbol{\delta}$ and the first i rows $[\mathbf{P}^{(1)} \;\; \mathbf{0}]$ of $\mathbf{P}$, with $\mathbf{P}^{(1)} \in \mathbb{K}[X]^{i\times i}$. A central property is that $\operatorname{amp}(\mathbf{s}^{(2)})$ is about half of $\operatorname{amp}(\mathbf{s})$ as well,

where $\mathbf{s}^{(2)}$ is the tail of $\mathbf{s}$ starting at the entry $i+1$. Note that this is not obvious, since we have $i \leqslant j$ with i unknown a priori.

Then, knowing the degrees $\boldsymbol{\delta}^{(1)}$ allows us to set up an approximant basis computation that yields a *residual*, that is, a column $\mathbf{G} \in \mathbb{K}[X]^{(m-i)\times 1}$ and a modulus $\mathfrak{n}$ such that we can continue the computation of $\mathbf{P}$ using a second recursive call which consists in computing the $\mathbf{s}^{(2)}$-Popov solution basis of $\mathrm{Syz}_{\mathfrak{n}}(\mathbf{G})$. From these two calls we obtain $\boldsymbol{\delta}$, and then $\mathbf{P}$ can be recovered efficiently using Algorithm 16.

Now we present the details of this approach and of its correctness. Let $\mathbf{F} \in \mathbb{K}[X]^{m\times 1}$, $\mathfrak{m} \in \mathbb{K}[X]_{\neq 0}$ with $D = \deg(\mathfrak{m}) > \deg(\mathbf{F})$, $\mathbf{s} \in \mathbb{Z}^m$, $\mathbf{P}$ be the $\mathbf{s}$-Popov solution basis of $\mathrm{Syz}_{\mathfrak{m}}(\mathbf{F})$, and $\boldsymbol{\delta}$ be the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_{\mathfrak{m}}(\mathbf{F})$. In what follows, $\boldsymbol{\pi}^{\mathbf{s}} = (\pi_1, \ldots, \pi_m)$ is any permutation of $\{1, \ldots, m\}$ such that $(s_{\pi_1}, \ldots, s_{\pi_m})$ is non-decreasing.

Then, for $\mathbf{t} = (t_1, \ldots, t_m) \in \mathbb{Z}^m$ we write $\mathbf{t}_{[i:j]}$ for the subtuple of $\mathbf{t}$ formed by its entries at indices $\{\pi_i, \ldots, \pi_j\}$, and for a matrix $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$ we write $\mathbf{A}_{[i:j,k:l]}$ for the submatrix of $\mathbf{A}$ formed by its rows at indices $\{\pi_i, \pi_{i+1}, \ldots, \pi_j\}$ and columns at indices $\{\pi_k, \pi_{k+1}, \ldots, \pi_l\}$. The main ideas in this section can be understood by focusing on the case of a non-decreasing $\mathbf{s}$, taking $\pi_i = i$ for all i: then we have $\mathbf{t}_{[i:j]} = (t_i, t_{i+1}, \ldots, t_j)$ and $\mathbf{A}_{[i:j,k:l]} = (\mathbf{A}_{u,v})_{i\leqslant u\leqslant j, k\leqslant v\leqslant l}$.

Remark 8.9. We recall that, by subtuple or submatrix we always mean that the respective positions of the rows are unchanged. For example, the subtuple of $(3, 7, 8)$ formed by its entries at indices $\{3, 1\}$ is $(3, 8)$.

Furthermore, one may wonder why we do not simply sort the shift $\mathbf{s}$ in non-decreasing order from the beginning, and permute the entries of $\mathbf{F}$ accordingly. The reason is that, if we find the shifted Popov basis for this permuted instance, then permuting back the columns of this basis do not necessarily yield the sought $\mathbf{s}$-Popov basis $\mathbf{P}$; while it does yield an $\mathbf{s}$-minimal basis, it is not clear how to deduce $\mathbf{P}$ efficiently from the latter. ☕

We now introduce the notion of splitting index, with the aim of helping us to locate zero blocks in the block-triangular shape of $\mathbf{P}$.

Definition 8.10 (Splitting index)**.** *Let* $\mathbf{d} = (d_i)_i \in \mathbb{Z}^m_{\geqslant 0}$, $\mathbf{t} \in \mathbb{Z}^m$, *and write* $(\mu_i)_i \in \mathbb{Z}^m_{>0}$ *the entries of* $\boldsymbol{\pi}^{\mathbf{t}}$. *Then, an integer* i *in* $\{1, \ldots, m-1\}$ *is said to be a* splitting index for $(\mathbf{d}, \mathbf{t})$ *if we have* $d_{\mu_j} + t_{\mu_j} - t_{\mu_{i+1}} < 0$ *for all* $j \in \{1, \ldots, i\}$.

We will see below in Lemma 8.12 that if i is a splitting index for $(\boldsymbol{\delta}, \mathbf{s})$, then the $\mathbf{s}$-Popov solution basis has a block triangular shape. Our algorithm first looks for such a splitting index, and then uses this shape to split the problem into two subproblems concerning the two diagonal blocks, which have of dimensions i and $m-i$. To find a splitting index, we rely on the following property.

Lemma 8.11. *Let* $\mathbf{d} \in \mathbb{Z}^m_{\geqslant 0}$ *and* $\mathbf{t} \in \mathbb{Z}^m$. *If* $(\mathbf{d}, \mathbf{t})$ *does not admit any splitting index, then we have* $|\mathbf{d}| \geqslant \mathrm{amp}(\mathbf{t})$.

Proof. Let us write $(\mu_i)_i \in \mathbb{Z}^m_{>0}$ the entries of $\boldsymbol{\pi}^{\mathbf{t}}$ and $(d_i)_i \in \mathbb{Z}^m_{\geqslant 0}$ the entries of $\mathbf{d}$. Since $m-1$ not a splitting index for $(\mathbf{d}, \mathbf{t})$, there exists $j_1 \in \{1, \ldots, m-1\}$ such that $d_{\mu_{j_1}} + t_{\mu_{j_1}} - t_{\mu_m} \geqslant 0$. Then, if $j_1 = 1$, we have $t_{\mu_{j_1}} - t_{\mu_m} = -\mathrm{amp}(\mathbf{t})$ and the property is proved; otherwise, since $j_1 - 1$ is not a splitting index, there exists $j_2 \in \{1, \ldots, j_1 - 1\}$ such that

$d_{\mu_{j_2}} + t_{\mu_{j_2}} - t_{\mu_{j_1}} \geqslant 0$. This way, we build $j_1 > j_2 > \cdots > j_r = 1$ with $d_{\mu_{j_k}} + t_{\mu_{j_k}} - t_{\mu_{j_{k-1}}} \geqslant 0$ for $1 \leqslant k \leqslant r$ (where we write $j_0 = m$): summing these inequalities gives the result. ☙

In the next result, we show that if i is a splitting index for $(\boldsymbol{\delta}, \mathbf{s})$, then the $\mathbf{s}$-Popov solution basis $\mathbf{P}$ has a block triangular shape up to row and column permutations:

$$\begin{bmatrix} \mathbf{P}_{[:i,:i]} & \mathbf{P}_{[:i,i+1:]} \\ \mathbf{P}_{[i+1:,:i]} & \mathbf{P}_{[i+1:,i+1:]} \end{bmatrix} = \begin{bmatrix} \mathbf{P}_{[:i,:i]} & \mathbf{0} \\ \mathbf{P}_{[i+1:,:i]} & \mathbf{P}_{[i+1:,i+1:]} \end{bmatrix};$$

in Section 8.2.2, we will exploit this shape to consider subproblems related to the diagonal blocks.

Lemma 8.12. *If i is a splitting index for $(\boldsymbol{\delta}, \mathbf{s})$, then $[\mathbf{P}_{[:i,:i]} \;\; \mathbf{P}_{[:i,i+1:]}] = [\mathbf{P}_{[:i,:i]} \;\; \mathbf{0}]$.*

Proof. Here, $\mathbf{P} = [p_{ij}]_{i,j}$ is in $\mathbf{s}$-Popov form with diagonal degrees $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m)$. This implies that the $\mathbf{s}$-pivot of the row i of $\mathbf{P}$ is on the diagonal and of degree δ_i. Thus, for all j, we have $\operatorname{rdeg}_{\mathbf{s}}(\mathbf{P}_{i,*}) = \delta_i + s_i \geqslant \deg(p_{ij}) + s_j$. Moreover, $\boldsymbol{\pi}^{\mathbf{s}} = (\pi_i)_i$ is such that $(s_{\pi_1}, \ldots, s_{\pi_m})$ is non-decreasing. Now, since i is a splitting index for $(\boldsymbol{\delta}, \mathbf{s})$, we have $\delta_{\pi_j} + s_{\pi_j} - s_{\pi_{i+1}} < 0$ for all $1 \leqslant j \leqslant i$. Then, for $k \geqslant i+1$, from the inequality above and $s_{\pi_k} \geqslant s_{\pi_{i+1}}$ we obtain that $\deg(p_{\pi_j,\pi_k}) \leqslant \delta_{\pi_j} + s_{\pi_j} - s_{\pi_{i+1}} < 0$, and therefore $p_{\pi_j,\pi_k} = 0$. This concludes the proof since $\mathbf{P}_{[:i,i+1:]}$ is precisely the submatrix of $\mathbf{P}$ formed by its coefficients at indices (π_j, π_k) for $j \leqslant i$ and $i+1 \leqslant k$. ☙

8.2.2 Fast algorithm for a single equation

Using the result above allows us to partition $\mathbf{s}$ into ℓ subtuples which all contain a splitting index, as follows.

Given $\alpha \in \mathbb{Z}_{>0}$, we let $\ell = 1 + \lfloor \operatorname{amp}(\mathbf{s})/\alpha \rfloor$ and we consider the subtuples $\mathbf{s}_1, \ldots, \mathbf{s}_\ell$ of $\mathbf{s}$ where $\mathbf{s}_k$ consists of the entries of $\mathbf{s}$ in $\{\min(\mathbf{s}) + (k-1)\alpha, \ldots, \min(\mathbf{s}) + k\alpha - 1\}$; this gives a subroutine $\textsc{Partition}(\mathbf{s}, \alpha) = (\mathbf{s}_1, \ldots, \mathbf{s}_\ell)$. Now we take $\alpha \geqslant 2D$ and we assume $s_{\pi_{i+1}} - s_{\pi_i} \leqslant D$ for $1 \leqslant i < m$ without loss of generality (see the end of Section 1.2.2). Then, for $1 \leqslant k < \ell$, since $|\boldsymbol{\delta}| \leqslant D$ according to Lemma 2.10, and since by construction $\operatorname{amp}(\mathbf{t}) > D$ with $\mathbf{t} = (\mathbf{s}_k, \min(\mathbf{s}_{k+1}))$, by the above remark $\mathbf{s}_k$ contains a splitting index for $(\boldsymbol{\delta}, \mathbf{s})$.

Still, we do not know in advance which entries of $\mathbf{s}_k$ correspond to splitting indices for $(\boldsymbol{\delta}, \mathbf{s})$. Thus we recursively compute the $\mathbf{s}$-Popov solution basis $\mathbf{P}^{(0)}$ for $\mathbf{s}_1, \ldots, \mathbf{s}_{\ell/2}$, and we are now going to prove that this gives us a splitting index which divides the computation into two subproblems, the first of which has been already solved by computing $\mathbf{P}^{(0)}$.

Lemma 8.13. *Let $j \in \{2, \ldots, m\}$, let $\mathbf{s}^{(0)} = \mathbf{s}_{[:j]}$, let $\mathbf{P}^{(0)}$ be the $\mathbf{s}^{(0)}$-Popov solution basis of $\operatorname{Syz}_{\mathfrak{m}}(\mathbf{F}_{[:j]})$, and let $\boldsymbol{\delta}^{(0)}$ be its $\mathbf{s}^{(0)}$-pivot degree. We assume that there is a splitting index i for $(\boldsymbol{\delta}^{(0)}, \mathbf{s}^{(0)})$ such that $i \leqslant j$. Let $\mathbf{P}^{(1)} \in \mathbb{K}[X]^{i \times i}$ be the $\mathbf{s}^{(1)}$-Popov solution basis of $\operatorname{Syz}_{\mathfrak{m}}(\mathbf{F}_{[:i]})$ with $\mathbf{s}^{(1)} = \mathbf{s}_{[:i]}$, and let $\boldsymbol{\delta}^{(1)}$ be its $\mathbf{s}^{(1)}$-pivot degree. Then i is a splitting index for $(\boldsymbol{\delta}, \mathbf{s})$ and we have $\mathbf{P}_{[:i,:i]} = \mathbf{P}^{(1)} = \mathbf{P}^{(0)}_{[:i,:i]}$; in particular, $\boldsymbol{\delta}_{[:i]} = \boldsymbol{\delta}^{(1)} = \boldsymbol{\delta}^{(0)}_{[:i]}$. In these identities, the entries of $\mathbf{P}^{(0)}$ and $\boldsymbol{\delta}^{(0)}$ are indexed by $\{\pi_1, \ldots, \pi_j\}$ sorted increasingly.*

Proof. Since i is a splitting index for $(\boldsymbol{\delta}^{(0)}, \mathbf{s}^{(0)})$ we have $[\mathbf{P}^{(0)}_{[:i,:i]} \;\; \mathbf{P}^{(0)}_{[:i,i+1:]}] = [\mathbf{Q} \;\; \mathbf{0}]$ for some $\mathbf{Q} \in \mathbb{K}[X]^{i\times i}$. Now, for any $\mathbf{B} \in \mathbb{K}[X]^{m\times m}$ with $[\mathbf{B}_{[:i,:i]} \;\; \mathbf{B}_{[:i,i+1:]}] = [\mathbf{P}^{(1)} \;\; \mathbf{0}]$, the submatrix $\mathbf{B}_{[:i,:]}$ is in $\mathbf{s}$-Popov form with its rows being solutions of $\mathrm{Syz}_{\mathfrak{m}}(\mathbf{F})$. Then, by minimality of $\mathbf{P}$, $\mathbf{P}_{[:i,:]}$ has $\mathbf{s}$-pivot degree at most $\boldsymbol{\delta}^{(1)}$ componentwise, so that i is also a splitting index for $(\boldsymbol{\delta}, \mathbf{s})$, and in particular $[\mathbf{P}_{[:i,:i]} \;\; \mathbf{P}_{[:i,i+1:]}] = [\mathbf{R} \;\; \mathbf{0}]$ for some $\mathbf{R} \in \mathbb{K}[X]^{i\times i}$. It remains to prove that $\mathbf{Q} = \mathbf{R} = \mathbf{P}^{(1)}$.

Since $\mathbf{R}\mathbf{F}_{[:i]} = 0 \bmod \mathfrak{m}$ and $\mathbf{R} = \mathbf{P}_{[:i,:i]}$ is in $\mathbf{s}^{(1)}$-Popov form, proving that all solutions $\mathbf{p} \in \mathbb{K}[X]^{1\times i}$ of $\mathrm{Syz}_{\mathfrak{m}}(\mathbf{F}_{[:i]})$ are in the row space of $\mathbf{R}$ is enough to obtain $\mathbf{R} = \mathbf{P}^{(1)}$. Since $\mathbf{q} \in \mathbb{K}[X]^{1\times m}$ defined by $[\mathbf{q}_{[:i]} \;\; \mathbf{q}_{[i+1:]}] = [\mathbf{p} \;\; \mathbf{0}]$ is a solution of $\mathrm{Syz}_{\mathfrak{m}}(\mathbf{F})$, $\mathbf{q} = \boldsymbol{\lambda}\mathbf{P}$ for some $\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m}$. Now $\mathbf{P}$ is nonsingular, thus $\mathbf{P}_{[:i,i+1:]} = \mathbf{0}$ implies that $[\boldsymbol{\lambda}_{[:i]} \;\; \boldsymbol{\lambda}_{[i+1:]}] = [\boldsymbol{\mu} \;\; \mathbf{0}]$ with $\boldsymbol{\mu} \in \mathbb{K}[X]^{1\times i}$, hence $\mathbf{p} = \mathbf{q}_{[:i]} = \boldsymbol{\lambda}_{[:i]}\mathbf{P}_{[:i,:i]} + \boldsymbol{\lambda}_{[i+1:]}\mathbf{P}_{[i+1:,:i]} = \boldsymbol{\mu}\mathbf{Q}$. Similar arguments give $\mathbf{Q} = \mathbf{P}^{(1)}$. ☙

The next two lemmas show that knowing $\boldsymbol{\delta}^{(1)}$, which is $\boldsymbol{\delta}_{[:i]}$, allows us to compute a so-called *residual* $(\mathfrak{n}, \mathbf{G})$ from which we can complete the computation of $\boldsymbol{\delta}$ and $\mathbf{P}$.

Lemma 8.14. *Let $\mathbf{s}^{(2)} = \mathbf{s}_{[i+1:]}$, let $\mathbf{d} = -\boldsymbol{\delta}^{(1)} + \min(\mathbf{s}^{(2)}) - 2D \in \mathbb{Z}^i$, let $\mathbf{v} \in \mathbb{Z}^m$ be such that $[\mathbf{v}_{[:i]} \;\; \mathbf{v}_{[i+1:]}] = [\mathbf{d} \;\; \mathbf{s}^{(2)}]$, and let $\mathbf{u} = (\mathbf{v}, \min(\mathbf{d})) \in \mathbb{Z}^{m+1}$. Besides, let $\left[\begin{smallmatrix} \mathbf{A} & \mathbf{q} \\ \mathbf{p} & q \end{smallmatrix}\right]$ denote the $\mathbf{u}$-Popov approximant basis at order $2D$ for $[\mathbf{F}^\mathsf{T} \;\; \mathfrak{m}]^\mathsf{T}$, where $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$ and $q \in \mathbb{K}[X]$. Then, we have $\deg(q) \geqslant D$, $\mathbf{A}_{[:i,i+1:]} = \mathbf{0}$, $\mathbf{p}_{[i+1:]} = \mathbf{0}$, and $[\mathbf{A}_{[:i,:i]} \;\; \mathbf{q}_{[:i]}] = [\mathbf{P}^{(1)} \;\; \mathbf{q}^{(1)}]$, where $\mathbf{q}^{(1)} = -\mathbf{P}^{(1)}\mathbf{F}_{[:i]}/\mathfrak{m}$.*

Proof. Since $\mathbf{u} = (\mathbf{v}, \min(\mathbf{v}))$ we have $\deg(\mathbf{p}) \leqslant \deg(q)$, and since $\deg(\mathbf{F}) < \deg(\mathfrak{m})$ the degree of $\mathbf{p}\mathbf{F} + q\mathfrak{m}$ is $\deg(q) + D$; then $\mathbf{p}\mathbf{F} + q\mathfrak{m} = 0 \bmod X^{2D}$ implies $\deg(q) + D \geqslant 2D$. Now, since $\mathbf{A}$ is in $\mathbf{v}$-Popov form and $\deg(\mathbf{A}) \leqslant 2D - \deg(q) < 2D$, from $\min(\mathbf{s}^{(2)}) \geqslant \max(\mathbf{d}) + 2D$ we get $\mathbf{A}_{[:i,i+1:]} = \mathbf{0}$. Besides, $\mathbf{p}_{[i+1:]} = \mathbf{0}$ since either $\deg(q) < 2D$ and then $\min(\mathbf{s}^{(2)}) > \min(\mathbf{d}) + \deg(q)$, or $\mathbf{A}$ is the identity matrix and then $\mathbf{p} = \mathbf{0}$.

Furthermore, by Lemma 8.2 $[\mathbf{P}^{(1)} \;\; \mathbf{q}^{(1)}]$ is the $(\mathbf{d}, \min(\mathbf{d}))$-Popov kernel basis for $[\mathbf{F}^\mathsf{T}_{[:i]} \;\; \mathfrak{m}]^\mathsf{T}$, with $(\mathbf{d}, \min(\mathbf{d}))$-pivot index $\{1, \ldots, i\}$, $(\mathbf{d}, \min(\mathbf{d}))$-pivot degree $\boldsymbol{\delta}^{(1)}$ and degree at most $\max(\boldsymbol{\delta}^{(1)})$. Then, following the arguments in the proof of Lemma 8.5, one can show that $[\mathbf{A}_{[:i,:i]} \;\; \mathbf{q}_{[:i]}] = [\mathbf{P}^{(1)} \;\; \mathbf{q}^{(1)}]$. ☙

Thus, up to row and column permutations this approximant basis is

$$\begin{bmatrix} \mathbf{A} & \mathbf{q} \\ \mathbf{p} & q \end{bmatrix} = \begin{bmatrix} \mathbf{P}^{(1)} & \mathbf{0} & \mathbf{q}^{(1)} \\ * & \mathbf{P}^{(2)} & * \\ * & \mathbf{0} & q \end{bmatrix}$$

with $\mathbf{P}^{(2)} = \mathbf{A}_{[i+1:,i+1:]} \in \mathbb{K}[X]^{(m-i)\times(m-i)}$ in $\mathbf{s}^{(2)}$-Popov form.

Lemma 8.15. *Let us denote by $\boldsymbol{\delta}^{(2)} \in \mathbb{Z}^{m-i}_{\geqslant 0}$ the $\mathbf{s}^{(2)}$-pivot degree of $\mathbf{P}^{(2)} = \mathbf{A}_{[i+1:,i+1:]}$, let $\mathfrak{n} = X^{-2D}(\mathbf{p}_{[:i]}\mathbf{F}_{[:i]} + q\mathfrak{m}) \in \mathbb{K}[X]$, and let $\mathbf{G} = X^{-2D}(\mathbf{A}_{[i+1:,:]}\mathbf{F} + \mathbf{q}_{[i+1:]}\mathfrak{m}) \in \mathbb{K}[X]^{(m-i)\times 1}$. Then, $\deg(\mathbf{G}) < \deg(\mathfrak{n}) \leqslant D - |\boldsymbol{\delta}^{(1)}| - |\boldsymbol{\delta}^{(2)}|$. Furthermore, let $\mathbf{P}^{(3)}$ be the $\mathbf{t}$-Popov solution basis of $\mathrm{Syz}_{\mathfrak{n}}(\mathbf{G})$, where $\mathbf{t} = \mathrm{rdeg}_{\mathbf{s}^{(2)}}(\mathbf{P}^{(2)})$, and denote by $\boldsymbol{\delta}^{(3)}$ its $\mathbf{t}$-pivot degree. Then, $(\boldsymbol{\delta}_{[:i]}, \boldsymbol{\delta}_{[i+1:]}) = (\boldsymbol{\delta}^{(1)}, \boldsymbol{\delta}^{(2)} + \boldsymbol{\delta}^{(3)})$.*

Proof. From Lemma 2.10, the sum $|\boldsymbol{\delta}^{(1)}| + |\boldsymbol{\delta}^{(2)}| + \deg(q)$ of the $\mathbf{u}$-pivot degrees of $\left[\begin{smallmatrix}\mathbf{A} & \mathbf{q}\\ \mathbf{p} & q\end{smallmatrix}\right]$ is at most the order $2D$. As a consequence, $\deg(\mathfrak{n}) = \deg(q) - D \leqslant D - |\boldsymbol{\delta}^{(1)}| - |\boldsymbol{\delta}^{(2)}|$, $\deg(\mathbf{A}_{[i+1:,:i]}) < |\boldsymbol{\delta}^{(1)}| \leqslant D$, $\deg(\mathbf{A}_{[i+1:,i+1:]}) \leqslant |\boldsymbol{\delta}^{(2)}| \leqslant D$, and $\deg(\mathbf{q}_{[i+1:]}) < \deg(q)$. This implies $\deg(\mathbf{G}) < \deg(q) - D = \deg(\mathfrak{n})$.

Let $\mathbf{q}^{(3)} = -\mathbf{P}^{(3)}\mathbf{G}/\mathfrak{n}$ and $t = \operatorname{rdeg}_{\mathbf{u}}([\mathbf{p} \;\; q]) = \deg(q) + \min(\mathbf{d}) \leqslant \min(\mathbf{s}^{(2)}) \leqslant \min(\mathbf{t})$. By Lemma 8.2, $[\mathbf{P}^{(3)} \;\; \mathbf{q}^{(3)}]$ is the $(\mathbf{t}, t)$-Popov kernel basis for $[\mathbf{G}^{\mathsf{T}} \;\; \mathfrak{n}]^{\mathsf{T}}$. Then, defining $\mathbf{B} \in \mathbb{K}[X]^{m\times m}$ and $\mathbf{c} \in \mathbb{K}[X]^{m\times 1}$ by

$$\begin{bmatrix} \mathbf{B}_{[:i,:i]} & \mathbf{B}_{[:i,i+1:]} & \mathbf{c}_{[:i]} \\ \mathbf{B}_{[i+1:,:i]} & \mathbf{B}_{[i+1:,i+1:]} & \mathbf{c}_{[i+1:]} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}^{(3)} & \mathbf{q}^{(3)} \end{bmatrix},$$

we have that the product

$$\begin{bmatrix} \mathbf{B} & \mathbf{c} \end{bmatrix} \begin{bmatrix} \mathbf{A} & \mathbf{q} \\ \mathbf{p} & q \end{bmatrix}$$

is a $\mathbf{u}$-minimal kernel basis for $[\mathbf{F}^{\mathsf{T}} \;\; \mathfrak{m}]^{\mathsf{T}}$, according to [ZLS12, Theorem 3.9]. As a result, Lemma 8.2 implies that

$$\bar{\mathbf{P}} = \begin{bmatrix} \mathbf{B} & \mathbf{c} \end{bmatrix} \begin{bmatrix} \mathbf{A} \\ \mathbf{p} \end{bmatrix}$$

is a $\mathbf{v}$-minimal solution basis of $\operatorname{Syz}_{\mathfrak{m}}(\mathbf{F})$.

It is easily checked that $\mathbf{P}$ is in $\mathbf{v}$-Popov form, so that the $\mathbf{v}$-Popov form of $\bar{\mathbf{P}}$ is $\mathbf{P}$ and the $\mathbf{v}$-pivot degree of $\bar{\mathbf{P}}$ is $\boldsymbol{\delta}$. Besides,

$$\begin{bmatrix} \bar{\mathbf{P}}_{[:i,:i]} & \bar{\mathbf{P}}_{[:i,i+1:]} \\ \bar{\mathbf{P}}_{[i+1:,:i]} & \bar{\mathbf{P}}_{[i+1:,i+1:]} \end{bmatrix} = \begin{bmatrix} \mathbf{P}^{(1)} & \mathbf{0} \\ \mathbf{P}^{(3)}\mathbf{A}_{2,1} + \mathbf{q}^{(3)}\mathbf{A}_{3,1} & \mathbf{P}^{(3)}\mathbf{P}^{(2)} \end{bmatrix};$$

thus the item (iv) of Theorem 1.28 implies that $(\boldsymbol{\delta}_{[:i]}, \boldsymbol{\delta}_{[i+1:]}) = (\boldsymbol{\delta}^{(1)}, \boldsymbol{\delta}^{(2)} + \boldsymbol{\delta}^{(3)})$. ☙

These results lead us to Algorithm 17. It takes as input the parameter α which dictates the amplitude of the subtuples that partition $\mathbf{s}$; as mentioned above, the initial call can be made with $\alpha = 2D$.

Proposition 8.16. *Algorithm 17 is correct, and if the input parameter is initially set to $\alpha = 2D$, it uses $\mathcal{O}\tilde{}(m^{\omega-1}D)$ operations in $\mathbb{K}$.*

Proof. The correctness follows from the results in this section.

At each base case of the recursion, we have in Step **1** the computation of the shifted Popov approximant basis of a column at order $\mathcal{O}(\alpha)$. By Proposition 7.10, denoting by m the dimension at this base case, this uses $\mathcal{O}\tilde{}(m^{\omega-1}\alpha)$ operations.

Running the algorithm with initial input $\alpha = 2D$, then the recursive tree has depth $\mathcal{O}(\log(\ell)) = \mathcal{O}(\log(1+\operatorname{amp}(\mathbf{s})/2D))$, with $\operatorname{amp}(\mathbf{s})/2D \in \mathcal{O}(m^2)$ (we recall that the entries of an arbitrary shift can be reduced a priori thanks to the bound D on the degree of the determinant of the output basis, see Section 1.2.2). All recursive calls are for a modulus of degree $D < \alpha$. The approximant basis computation at Step **2.b** then uses $\mathcal{O}\tilde{}(m^{\omega-1}D)$ operations. Besides, the computation of $\mathbf{G}$ and $\mathfrak{n}$ at Step **2.c** can be done in time $\mathcal{O}\tilde{}(m^{\omega-1}D)$ using partial linearization as in Lemma 6.4, and Step **2.e** uses $\mathcal{O}\tilde{}(m^{\omega-1}D)$ operations by Proposition 8.7.

Algorithm 17 – FastPopovSolBasOneEq
(Solution basis for a single equation)

Input:

- a polynomial $\mathfrak{m} \in \mathbb{K}[X]_{\neq 0}$ of degree D,
- a column $\mathbf{F} \in \mathbb{K}[X]^{m\times 1}$ with $\deg(\mathbf{F}) < D$,
- a shift $\mathbf{s} \in \mathbb{Z}^m$,
- a parameter $\alpha \in \mathbb{Z}_{>0}$ with $\alpha \geqslant 2D$.

Output: the $\mathbf{s}$-Popov solution basis of $\mathrm{Syz}_{\mathfrak{m}}(\mathbf{F})$.

1. If $\mathrm{amp}(\mathbf{s}) \leqslant 2\alpha$:

a. $\mathbf{A} \leftarrow$ FastPopovAppBas$(2\alpha + 2D, [\mathbf{F}^\mathsf{T} \;\; \mathfrak{m}]^\mathsf{T}, (\mathbf{s}, \min(\mathbf{s})))$

b. Return the principal $m \times m$ submatrix of $\mathbf{A}$

2. Else: /* $\ell = 1 + \lfloor \mathrm{amp}(\mathbf{s})/\alpha \rfloor \geqslant 3$ */

a. $(\mathbf{s}_1, \ldots, \mathbf{s}_\ell) \leftarrow$ Partition$(\mathbf{s}, \alpha)$,
$j \leftarrow$ sum of the lengths of $\mathbf{s}_1, \ldots, \mathbf{s}_{\lceil \ell/2 \rceil}$, $\mathbf{s}^{(0)} \leftarrow \mathbf{s}_{[:j]}$,
$\mathbf{P}^{(0)} \leftarrow$ FastPopovSolBasOneEq$(\mathfrak{m}, \mathbf{F}_{[:j]}, \mathbf{s}^{(0)}, \alpha)$
$\boldsymbol{\delta}^{(0)} \leftarrow$ diagonal degrees of $\mathbf{P}^{(0)}$

b. $i \leftarrow$ the largest splitting index for $(\boldsymbol{\delta}^{(0)}, \mathbf{s}^{(0)})$, $\boldsymbol{\delta}^{(1)} \leftarrow \boldsymbol{\delta}^{(0)}_{[:i]}$,
$\mathbf{s}^{(2)} \leftarrow \mathbf{s}_{[i+1:]}$, $\mathbf{d} \leftarrow -\boldsymbol{\delta}^{(1)} + \min(\mathbf{s}^{(2)}) - 2D$,
$\mathbf{v} \in \mathbb{Z}^m$ with $[\mathbf{v}_{[:i]} \;\; \mathbf{v}_{[i+1:]}] \leftarrow [\mathbf{d} \;\; \mathbf{s}^{(2)}]$, $\mathbf{u} \leftarrow (\mathbf{v}, \min(\mathbf{d}))$
$\begin{bmatrix} \mathbf{A} & \mathbf{q} \\ \mathbf{p} & q \end{bmatrix} \leftarrow$ FastPopovAppBas$(2D, [\mathbf{F}^\mathsf{T} \;\; \mathfrak{m}]^\mathsf{T}, \mathbf{u})$

c. $\boldsymbol{\delta}^{(2)} \leftarrow$ the $\mathbf{s}^{(2)}$-pivot degree of $\mathbf{A}_{[i+1:,i+1:]}$
$\mathbf{G} \leftarrow X^{-2D}(\mathbf{A}_{[i+1:,:]}\mathbf{F} + \mathbf{q}_{[i+1:]}\mathfrak{m})$
$\mathfrak{n} \leftarrow X^{-2D}(\mathbf{p}_{[:i]}\mathbf{F}_{[:i]} + q\mathfrak{m})$

d. $\mathbf{t} \leftarrow \mathbf{s}^{(2)} + \boldsymbol{\delta}^{(2)} = \mathrm{rdeg}_{\mathbf{s}^{(2)}}(\mathbf{A}_{[i+1:,i+1:]})$
$\mathbf{P}^{(3)} \leftarrow$ FastPopovSolBasOneEq$(\mathfrak{n}, \mathbf{G}, \mathbf{t}, \alpha)$
$\boldsymbol{\delta}^{(3)} \leftarrow$ diagonal degrees of $\mathbf{P}^{(3)}$

e. $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$ with $(\boldsymbol{\delta}_{[:i]}, \boldsymbol{\delta}_{[i+1:]}) \leftarrow (\boldsymbol{\delta}^{(1)}, \boldsymbol{\delta}^{(2)} + \boldsymbol{\delta}^{(3)})$
Return MinDegSolBas$(\mathfrak{m}, \mathbf{F}, \mathbf{s}, \boldsymbol{\delta})$

On a given level of the tree, the sum of the dimensions of the column vector in input of each subproblem is in $\mathcal{O}(m)$. Since $a^{\omega-1} + b^{\omega-1} \leqslant (a+b)^{\omega-1}$ for all $a, b > 0$, each level of the tree uses a total of $\tilde{\mathcal{O}}(m^{\omega-1}\alpha)$ operations. ☙

8.3 Fast solution bases in Popov form for arbitrary shifts

Now that we have an efficient algorithm for the case of one equation $n = 1$, our main algorithm uses a divide-and-conquer approach on n. For efficiency, we follow ideas described in Section 1.2.1 and already used above in Algorithm 14: from the two bases obtained recursively, we deduce the $\mathbf{s}$-minimal degree $\boldsymbol{\delta}$; then, knowing $\boldsymbol{\delta}$, we compute the sought solution basis via Algorithm 16.

When the dimension $D = \deg(\mathfrak{m}_1) + \cdots + \deg(\mathfrak{m}_n)$ is small, namely at most m, we rely on the general relation basis algorithm detailed in Section 2.2 and Chapter 4. For the computation of the *residual* $\mathbf{G}$ at Step **3.c**, we use partial linearization as detailed in Lemma 6.4 in Section 6.3.

Proposition 8.17. *Algorithm 18 is correct; assuming $n \in \mathcal{O}(m)$, it uses $\tilde{\mathcal{O}}(m^{\omega-1}D)$ operations in $\mathbb{K}$.*

Proof. The correctness and the cost $\tilde{\mathcal{O}}(m^{\omega-1}D)$ for Steps **1** and **2** follow from Proposition 4.18 and Proposition 8.16. With the costs of Steps **3.c** and **3.e** given in Proposition 8.7 and Lemma 6.4, we obtain the announced cost bound.

Concerning the correctness Step **3**, we use Theorem 1.28 with the module $\mathcal{M}^{(1)}$ of solutions of $\mathrm{Syz}_{\mathfrak{M}^{(1)}}(\mathbf{F}^{(1)})$ and the module $\mathcal{M}^{(2)}$ of solutions of $\mathrm{Syz}_{\mathfrak{M}^{(2)}}(\mathbf{F}^{(2)})$. The item (iv) of this theorem implies that the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ is the sum $\boldsymbol{\delta}^{(1)} + \boldsymbol{\delta}^{(2)}$, and therefore Step **3.h** computes the $\mathbf{s}$-Popov solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ according to Proposition 8.7.

As for a cost bound, we have two recursive calls at Steps **3.c** and **3.f**, the computation of the residual at Step **3.e**, and the computation of the output basis at Step **3.h**. The first recursive call involves the first $\lfloor n/2 \rfloor$ equations $\mathbf{F}^{(1)}$ modulo $\mathfrak{M}^{(1)}$, with the sum of the degrees of the moduli being $D^{(1)} = D_1 + \cdots + D_{\lfloor n/2 \rfloor}$, while the second recursive call is for the updated remaining $\lceil n/2 \rceil$ equations $\mathbf{G}$ modulo $\mathfrak{M}^{(2)}$, with the sum of the degrees of the moduli being $D^{(2)} = D_1 + \cdots + D_{\lceil n/2 \rceil}$. Although $D^{(1)}$ and $D^{(2)}$ may be unbalanced, we have $D = D^{(1)} + D^{(2)}$: if the cost of Steps **3.e** and **3.h** is in $\tilde{\mathcal{O}}(m^{\omega-1}D)$, then this implies that the overall cost is in $\tilde{\mathcal{O}}(m^{\omega-1}D)$ as well.

Proposition 8.7 states that the call to MinDegSolBas at Step **3.h** uses $\tilde{\mathcal{O}}(m^{\omega-1}D)$. Besides, according to Lemma 6.4, the same cost bound holds concerning the computation of $\mathbf{G}$ at Step **3.e**. Indeed, by Lemma 2.10 the sum of column degrees of $\mathbf{P}^{(1)}$ is at most $D^{(1)} \leqslant D$, the sum of the degrees of the moduli in $\mathfrak{M}^{(2)}$ is at most $D^{(2)} \leqslant D$, we have $m \leqslant D$, and $n \in \mathcal{O}(m)$ by assumption. ☙

Algorithm 18 – FastPopovSolBas
(Shifted Popov solution basis)

Input:

- polynomials $\mathfrak{M} = (\mathfrak{m}_1, \ldots, \mathfrak{m}_n) \in \mathbb{K}[X]^n_{\neq 0}$ of degrees $(D_1, \ldots, D_n)$,
- a matrix $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ with $\deg(\mathbf{F}_{*,j}) < D_j$ for $1 \leqslant j \leqslant n$,
- a shift $\mathbf{s} \in \mathbb{Z}^m$.

Output: the $\mathbf{s}$-Popov solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$.

1. If $D = D_1 + \cdots + D_n \leqslant m$:

a. For j from 1 to n:

(i) write $\mathbf{F}_{*,j} = \mathbf{f}_j^{(0)} + \mathbf{f}_j^{(1)} X + \cdots + \mathbf{f}_j^{(D_j-1)} X^{D_j-1}$
and $\mathfrak{m}_j = c_j^{(0)} + c_j^{(1)} X + \cdots + c_j^{(D_j-1)} X^{D_j-1} + X^{D_j}$

(ii) $\mathbf{E}_j \leftarrow \begin{bmatrix} \mathbf{f}_j^{(0)} & \mathbf{f}_j^{(1)} & \cdots & \mathbf{f}_j^{(D_j-1)} \end{bmatrix} \in \mathbb{K}^{m \times D_j}$

(iii) $\mathbf{C}_j \leftarrow \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & 0 & 1 \\ -c_j^{(0)} & -c_j^{(1)} & \cdots & -c_j^{(D_j-1)} \end{bmatrix} \in \mathbb{K}^{D_j \times D_j}$

b. $\mathbf{E} \leftarrow \begin{bmatrix} \mathbf{E}_1 & \cdots & \mathbf{E}_n \end{bmatrix} \in \mathbb{K}^{m \times D}$; $\mathbf{C} \leftarrow \mathrm{diag}(\mathbf{C}_1, \ldots, \mathbf{C}_n) \in \mathbb{K}^{D \times D}$

c. Return LinPopovRelBas$(\mathbf{C}, \mathbf{E}, \mathbf{s})$

2. Else if $n = 1$ then return FastPopovSolBasOneEq$(\mathfrak{m}_1, \mathbf{F}, \mathbf{s}, 2D)$

3. Else:

a. $\mathfrak{M}^{(1)} \leftarrow (\mathfrak{m}_1, \ldots, \mathfrak{m}_{\lfloor n/2 \rfloor})$; $\mathbf{F}^{(1)} \leftarrow \mathbf{F}_{*,1\ldots\lfloor n/2 \rfloor}$

b. $\mathfrak{M}^{(2)} \leftarrow (\mathfrak{m}_{\lfloor n/2 \rfloor+1}, \ldots, \mathfrak{m}_n)$; $\mathbf{F}^{(2)} \leftarrow \mathbf{F}_{*,\lfloor n/2 \rfloor+1\ldots n}$

c. $\mathbf{P}^{(1)} \leftarrow$ FastPopovSolBas$(\mathfrak{M}^{(1)}, \mathbf{F}^{(1)}, \mathbf{s})$

d. $\boldsymbol{\delta}^{(1)} \leftarrow$ diagonal degrees of $\mathbf{P}^{(1)}$

e. $\mathbf{G} \leftarrow \mathbf{P}^{(1)} \mathbf{F}^{(2)} \bmod \mathfrak{M}^{(2)}$

f. $\mathbf{P}^{(2)} \leftarrow$ FastPopovSolBas$(\mathfrak{M}^{(2)}, \mathbf{G}, \mathbf{s} + \boldsymbol{\delta}^{(1)})$

g. $\boldsymbol{\delta}^{(2)} \leftarrow$ diagonal degrees of $\mathbf{P}^{(2)}$

h. Return MinDegSolBas$(\mathfrak{M}, \mathbf{F}, \mathbf{s}, \boldsymbol{\delta}^{(1)} + \boldsymbol{\delta}^{(2)})$

9

Computing a solution via structured linear algebra

In this chapter, we focus on Problem 10, which asks to compute a degree-constrained solution to a system of linear modular equations over $\mathbb{K}[X]$. A natural approach is to rely on the algorithm in Chapter 8, and retrieve a degree-constrained solution as being the row with smallest degree in a full shifted minimal basis of solutions. Here, using fast algorithms for solving structured linear systems over $\mathbb{K}$, we give a probabilistic approach which is slightly faster. A detailed overview of this approach and a comparison with previous work were presented in Section 2.5.

We start by summarizing some general aspects of solving structured linear systems (Section 9.1). Then, in Section 9.2, we will present a first solution based on a reduction to the case of moduli that are powers of X, leading to the consideration of a mosaic-Hankel system over $\mathbb{K}$. Finally, in Section 9.3 we give another solution, which directly rewrites the modular equations over $\mathbb{K}[X]$ as a linear system over $\mathbb{K}$ with a Toeplitz-like structure.

9.1 Solving structured homogeneous linear systems

Our two solutions to Problem 10 rely on fast algorithms for solving linear systems of the form $\mathbf{A}\mathbf{u} = 0$ with $\mathbf{A}$ a structured matrix over $\mathbb{K}$. In this section, we briefly review useful concepts and results related to *displacement rank* techniques. While these techniques can handle systems with several kinds of structure, we will only need (and discuss) those related to *Toeplitz-like* and *Hankel-like* systems; for a more comprehensive treatment, the reader may consult [Pan01].

Let m be a positive integer and let $\mathbf{Z}_m \in \mathbb{K}^{m\times m}$ be the square matrix with ones on the subdiagonal and zeros elsewhere:

$$\mathbf{Z}_m = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix} \in \mathbb{K}^{m\times m}.$$

Given two integers m and n, consider the following operators:

$$\begin{array}{rccc} \Delta_{m,n}: & \mathbb{K}^{m\times n} & \to & \mathbb{K}^{m\times n} \\ & \mathbf{A} & \mapsto & \mathbf{A} - \mathbf{Z}_m \mathbf{A} \mathbf{Z}_n^{\mathsf{T}} \end{array}$$

and

$$\begin{array}{rccc} \Delta'_{m,n}: & \mathbb{K}^{m\times n} & \to & \mathbb{K}^{m\times n} \\ & \mathbf{A} & \mapsto & \mathbf{A} - \mathbf{Z}_m \mathbf{A} \mathbf{Z}_n, \end{array}$$

which subtract from $\mathbf{A}$ its translate one place along the diagonal and the anti-diagonal, respectively.

Let us discuss $\Delta_{m,n}$ first. If $\mathbf{A}$ is a *Toeplitz* matrix, that is, invariant along diagonals, $\Delta_{m,n}(\mathbf{A})$ has rank at most two. As it turns out, Toeplitz systems can be solved much faster than general linear systems, in quasi-linear time in $m+n$. The main idea behind algorithms for structured matrices is to extend these algorithmic properties to those matrices $\mathbf{A}$ for which the rank of $\Delta_{m,n}(\mathbf{A})$ is small, in which case we say that $\mathbf{A}$ is *Toeplitz-like*. Below, this rank will be called the *displacement rank* of $\mathbf{A}$ (with respect to $\Delta_{m,n}$).

A pair of matrices $(\mathbf{V}, \mathbf{W})$ in $\mathbb{K}^{m\times\alpha} \times \mathbb{K}^{\alpha\times n}$ will be called a *generator of length* α for $\mathbf{A}$ with respect to $\Delta_{m,n}$ if $\Delta_{m,n}(\mathbf{A}) = \mathbf{VW}$. For the structure we are considering, one can recover $\mathbf{A}$ from its generator; in particular, one can use a generator of length α as a way to represent $\mathbf{A}$ using $\alpha(m+n)$ field elements. One of the main aspects of structured linear algebra algorithms is to use generators as a compact data structure throughout the whole process.

Up to now, we only discussed the Toeplitz structure. *Hankel-like* matrices are those which have a small displacement rank with respect to $\Delta'_{m,n}$, that is, those matrices $\mathbf{A}$ for which the rank of $\Delta'_{m,n}(\mathbf{A})$ is small. As far as solving the system $\mathbf{Au} = 0$ is concerned, this case can easily be reduced to the Toeplitz-like case. Define $\mathbf{B} = \mathbf{AJ}_n$, where $\mathbf{J}_n$ is the reversal matrix of size n, all entries of which are zero, except the anti-diagonal which is set to one. Then, one easily checks that the displacement rank of $\mathbf{A}$ with respect to $\Delta'_{m,n}$ is the same as the displacement rank of $\mathbf{B}$ with respect to $\Delta_{m,n}$, and that if $(\mathbf{V}, \mathbf{W})$ is a generator for $\mathbf{A}$ with respect to $\Delta'_{m,n}$, then $(\mathbf{V}, \mathbf{WJ}_n)$ is a generator for $\mathbf{B}$ with respect to $\Delta_{m,n}$. Using the algorithm for Toeplitz-like matrices gives us a solution $\mathbf{v}$ to $\mathbf{Bv} = 0$, from which we deduce that $\mathbf{u} = \mathbf{J}_n\mathbf{v}$ is a solution to $\mathbf{Au} = 0$.

In this thesis, we will not enter the details of algorithms for solving such structured systems. The main result we will rely on is the next proposition; this result can be found in [BJS08, BJMS16], which features the best known complexity for this kind of task, to the best of our knowledge. This algorithm is based on previous work of Bitmead and Anderson [BA80], Morf [Mor80], Kaltofen [Kal94], and Pan [Pan01], and is probabilistic (it depends on the choice of some parameters in the base field $\mathbb{K}$, and success is ensured provided these parameters avoid a hypersurface of the parameter space).

The proof of the following proposition occupies the rest of this section. Remark that some aspects of this statement could be improved (for instance, we could reduce the cost so that it only depends on m, not $\max(m, n)$), but that would be inconsequential for the applications we make of it.

Proposition 9.1. *Given a generator* $(\mathbf{V}, \mathbf{W})$ *of length* α *for a matrix* $\mathbf{A} \in \mathbb{K}^{m\times n}$*, with respect to either* $\Delta_{m,n}$ *or* $\Delta'_{m,n}$*, one can find a nonzero element in the right nullspace of* $\mathbf{A}$*,*

or determine that none exists, by a probabilistic algorithm that uses $\mathcal{O}(\alpha^{\omega-1}\mathsf{M}(D)\log(D))$ *operations in* $\mathbb{K}$*, with* $D = \max(m, n)$*. The algorithm chooses* $\mathcal{O}(D)$ *elements in* $\mathbb{K}$*; if these elements are chosen uniformly at random in a subset of* $\mathbb{K}$ *of cardinality at least* $6D^2$*, the probability of success is at least* $1/2$*.*

Square matrices. In all that follows, we consider only the operator $\Delta_{m,n}$, since we already pointed out that the case of $\Delta'_{m,n}$ can be reduced to it for no extra cost.

When $m = n$, we use directly the main results of [BJS08, BJMS16], which give the running time reported above. However, they do not explicitly state which solution we obtain, as they are written for general non-homogeneous systems. Here, we want to make sure that we obtain a nonzero element in the right nullspace (if one exists), so slightly more details are needed.

The algorithm in that theorem chooses $3m-2$ elements in $\mathbb{K}$, the first $2m-2$ of which are used to precondition $\mathbf{A}$ by giving it generic rank profile; this is the case when these parameters avoid a hypersurface of $\mathbb{K}^{2m-2}$ of degree at most $m^2 + m$.

Assume this is the case. Then, following [KS91], the output vector $\mathbf{u}$ is obtained in a parametric form as $\mathbf{u} = \ell(\mathbf{u}')$, where $\mathbf{u}'$ consists of another set of m parameters chosen in $\mathbb{K}$ and ℓ is a surjective linear mapping with image the right nullspace $\ker(\mathbf{A})$ of $\mathbf{A}$. If $\ker(\mathbf{A})$ is trivial, the algorithm returns the zero vector in any case, which is correct. Otherwise, the set of vectors $\mathbf{u}'$ such that $\ell(\mathbf{u}') = 0$ is contained in a hyperplane of $\mathbb{K}^m$, so it is enough to choose $\mathbf{u}'$ outside of that hyperplane to ensure success.

To conclude we rely on the Zippel-Schwartz lemma [DL78, Zip79, Sch80], which can be summarized as follows: if a nonzero polynomial over $\mathbb{K}$ of total degree at most d is evaluated by assigning each of its indeterminates a value chosen uniformly at random in a subset S of $\mathbb{K}$, then the probability that the resulting polynomial value be zero is at most $d/\mathrm{Card}(S)$. Thus, applying that result to the polynomial of degree $d := m^2+m+1 \leqslant 3m^2$ corresponding to the hypersurface and the hyperplane mentioned above, we see that if we choose all parameters uniformly at random in a subset $S \subseteq \mathbb{K}$ of cardinality $\mathrm{Card}(S) \geqslant 6m^2$, the algorithm succeeds with probability at least $1/2$.

Wide matrices. Suppose now that $m < n$, so that the system is underdetermined. We add $n - m$ zero rows on top of $\mathbf{A}$, obtaining an $n \times n$ matrix $\mathbf{B}$. Applying the algorithm for the square case to $\mathbf{B}$, we will obtain a right nullspace element $\mathbf{u}$ for $\mathbf{B}$ and thus for $\mathbf{A}$, since these nullspaces are the same. In order to do so, we need to construct a generator for $\mathbf{B}$ from the generator $(\mathbf{V}, \mathbf{W})$ we have for $\mathbf{A}$: one simply takes $(\hat{\mathbf{V}}, \mathbf{W})$, where $\hat{\mathbf{V}}$ is the matrix in $\mathbb{K}^{n\times\alpha}$ obtained by adding $n - m$ zero rows on top of $\mathbf{V}$.

Tall matrices. Suppose finally that $m > n$. This time, we build the matrix $\mathbf{B} \in \mathbb{K}^{m\times m}$ by adjoining $m - n$ zero columns to $\mathbf{A}$ on the left. The generator $(\mathbf{V}, \mathbf{W})$ of $\mathbf{A}$ can be turned into a generator of $\mathbf{B}$ by simply adjoining $m - n$ zero columns to $\mathbf{W}$ on the left. We then solve the system $\mathbf{Bv} = 0$, and return the vector $\mathbf{u}$ obtained by discarding the first $m - n$ entries of $\mathbf{v}$.

The cost of this algorithm fits into the requested bound; all that remains to see is that we obtain a nonzero vector in the right nullspace $\ker(\mathbf{A})$ of $\mathbf{A}$ with nonzero probability.

Indeed, the nullspaces of $\mathbf{A}$ and $\mathbf{B}$ are now related by the equality $\ker(\mathbf{B}) = \mathbb{K}^{m-n} \times \ker(\mathbf{A})$. We mentioned earlier that in the algorithm for the square case, the solution $\mathbf{v}$ to $\mathbf{B}\mathbf{v} = 0$ is obtained in parametric form, as $\mathbf{v} = \ell(\mathbf{v}')$ for $\mathbf{v}' \in \mathbb{K}^m$, with ℓ a surjective mapping $\mathbb{K}^m \to \ker(\mathbf{B})$. Composing with the projection $\pi : \ker(\mathbf{B}) \to \ker(\mathbf{A})$, we obtain a parametrization of $\ker(\mathbf{A})$ as $\mathbf{u} = (\pi \circ \ell)(\mathbf{v}')$. The error probability analysis is then the same as in the square case.

9.2 Reducing to solving a mosaic-Hankel linear system

In this section, we give our first solution to Problem 10, thereby proving Theorem 2.25; this solution is outlined in Algorithm 19. It consists in first deriving equations modulo powers of X as in Lemma 9.3 below, and then linearizing them to obtain a mosaic-Hankel linear system which can be solved using the approach recalled in Section 9.1.

The derivation of the equations in Lemma 9.3 generalizes ideas from [RR00, ZGA11, Zeh13]. In these references, one focuses on solving the multivariate interpolation problem Problem 11 using the reduction to Problem 10 given in Section 11.1; in this context, these modular equations are some type of *extended key equations* [RR00, ZGA11, Zeh13].

We first recall Problem 10. We have in input some moduli $\mathfrak{M} = (\mathfrak{m}_0, \ldots, \mathfrak{m}_{n-1}) \in \mathbb{K}[X]^n_{\neq 0}$, and a matrix $\mathbf{F} \in \mathbb{K}[X]^{n \times m}$. Denoting by D_i the degree of $\mathfrak{m}_i$, we assume that the entries of the row $\mathbf{F}_{i,*}$ have degree less than D_i, for $0 \leqslant i < n$.

Remark 9.2. In this chapter, we consider the equations as being given by the rows of $\mathbf{F}$, and not by its columns as we did until now; this is because we will base our algorithms on structured linear system solving, for which it is more common to see the unknown as a column vector on the right of the system matrix. Furthermore, in this chapter the row and column indices of matrices will be numbered starting from 0. ☕

Then, given degree bounds forming a shift $\mathbf{N} = (-N_0, \ldots, -N_{m-1}) \in \mathbb{Z}^m_{<0}$, we look for a column vector $\mathbf{p} \in \mathbb{K}[X]^{m \times 1}$ such that the following holds:

(a) $\mathbf{p}$ is nonzero,

(b) $\operatorname{cdeg}_{\mathbf{N}}(\mathbf{p}) < 0$,

(c) $\mathbf{F}\mathbf{p} = 0 \bmod \mathfrak{M}$.

We recall that, writing $\mathbf{p} = [p_j]_{1 \leqslant j \leqslant m}$, the item *(b)* is equivalent to having $\deg(p_j) < N_j$ for all $0 \leqslant j < m$, while *(c)* is equivalent to having $\mathbf{F}_{i,*}\,\mathbf{p} = 0 \bmod \mathfrak{m}_i$ for all $0 \leqslant i < n$. In this section, without loss of generality, we assume that the moduli in $\mathfrak{M}$ are monic.

Our goal here is to linearize the condition *(c)* into a homogeneous linear system over $\mathbb{K}$ involving D linear equations with N unknowns, where $D = D_0 + \cdots + D_{n-1}$ and $N = N_0 + \cdots + N_{m-1}$. Without loss of generality, we will assume that

$$N \leqslant D + 1. \tag{9.1}$$

Indeed, if $N \geqslant D + 1$, then the instance of Problem 10 we are considering has more unknowns than equations and we may set the last $N - (D + 1)$ unknowns to zero while keeping the system underdetermined. This simply amounts to replacing the degree bounds

$N_0, \ldots, N_{m-1}$ by $N_0, \ldots, N_{m'-2}, N'_{m'-1}$, for $m' \leqslant m$ and $N'_{m'-1} \leqslant N_{m'-1}$ such that $N_0 + \cdots + N_{m'-2} + N'_{m'-1} = D + 1$. In particular, m may only decrease through this process.

In what follows, we write $\mathbf{F} = [F_{i,j}]_{i,j}$ and we will work with the reversals of the input and output polynomials, defined by

$$\begin{aligned}\overline{\mathfrak{m}_i} &= X^{D_i}\mathfrak{m}_i(X^{-1}),\\ \overline{F_{i,j}} &= X^{D_i-1}F_{i,j}(X^{-1}),\\ \overline{p_j} &= X^{N_j-1}p_j(X^{-1}).\end{aligned}$$

Let also $\beta = \max_{h<m} N_h$ and, for $0 \leqslant i < n$ and $0 \leqslant j < m$,

$$\delta_i = D_i + \beta - 1 \quad \text{and} \quad \gamma_j = \beta - N_j.$$

In particular, $\delta_i > 0$ and $\gamma_j \geqslant 0$; recalling that $\mathfrak{m}_i$ is monic, we can define further the polynomials $S_{i,j}$ in $\mathbb{K}[X]$ as

$$S_{i,j} = \frac{X^{\gamma_j}\overline{F_{i,j}}}{\overline{\mathfrak{m}_i}} \bmod X^{\delta_i}$$

for $0 \leqslant i < n$ and $0 \leqslant j < m$. (Those polynomials can be seen as a generalization of the ones that are usually called *syndrome polynomials* in the context of coding theory; see for example [ZGA11].) By using these polynomials, we can now reformulate the approximation condition of Problem 10 in terms of a set of *extended key equations* [ZGA11]:

Lemma 9.3. *Let* $\mathbf{p} = [p_j]_j \in \mathbb{K}[X]^{m\times 1}$ *satisfy the degree constraints in* (b). *Then,* $\mathbf{p}$ *satisfies the condition* (c) *if and only if for all* $i \in \{0, \ldots, n-1\}$, *there exists a polynomial* $T_i \in \mathbb{K}[X]$ *such that*

$$\sum_{0\leqslant j<m} S_{i,j}\overline{p_j} = T_i \bmod X^{\delta_i} \quad \textit{and} \quad \deg(T_i) < \beta - 1. \tag{9.2}$$

Proof. The condition *(c)* holds if and only if for all i in $\{0, \ldots, n-1\}$, there exists a polynomial $B_i \in \mathbb{K}[X]$ such that

$$\sum_{0\leqslant j<m} F_{i,j}p_j = B_i\mathfrak{m}_i. \tag{9.3}$$

For all i, j, the summand $F_{i,j}p_j$ has degree less than D_i+N_j-1, so the left-hand term above has degree less than δ_i. Since $\mathfrak{m}_i$ has degree D_i, this implies that whenever a polynomial B_i as above exists, we must have $\deg(B_i) < \delta_i - D_i = \beta - 1$. Now, by substituting $1/X$ for X and multiplying by X^{δ_i-1} we can rewrite the identity in (9.3) as

$$\sum_{0\leqslant j<m} \overline{F_{i,j}}\overline{p_j}X^{\gamma_j} = T_i\overline{\mathfrak{m}_i}, \tag{9.4}$$

where T_i is the polynomial of degree less than $\beta - 1$ given by $T_i = X^{\beta-2}B_i(X^{-1})$. Since the degrees of both sides of (9.4) are less than δ_i, one can consider the above identity modulo X^{δ_i} without loss of generality, and since $\overline{\mathfrak{m}_i}(0) = 1$ one can further divide by $\overline{\mathfrak{m}_i}$ modulo X^{δ_i}. This shows that Eq. (9.4) is equivalent to the identity in Eq. (9.2), and the proof is complete. ∎

Following [RR00, ZGA11], we are going to rewrite the conditions in Eq. (9.2) as a linear system in the coefficients of the polynomials $p_0, \ldots, p_{m-1}$, eliminating the unknowns T_i from the outset. We first define the *coefficient vector* of a solution $\mathbf{p} = [p_0, \ldots, p_{m-1}]$ to Problem 10 as the vector in $\mathbb{K}^{N\times 1}$ obtained by concatenating the vectors $\left[p_j^{(0)}, p_j^{(1)}, \ldots, p_j^{(N_j-1)}\right]^\mathsf{T}$ of the coefficients of p_j, for $0 \leqslant j < m$. Furthermore, we set up the block matrix

$$\mathbf{H} = \left[\mathbf{H}_{i,j}\right]_{0\leqslant i<n, 0\leqslant j<m} \in \mathbb{K}^{D\times N},$$

whose block (i, j) is the Hankel matrix

$$\mathbf{H}_{i,j} = \left[S_{i,j}^{(u+v+\gamma_j)}\right]_{0\leqslant u<D_i, 0\leqslant v<N_j} \in \mathbb{K}^{D_i\times N_j},$$

where $S_{i,j}^{(0)}, S_{i,j}^{(1)}, \ldots, S_{i,j}^{(\delta_i-1)}$ denote the $\delta_i \geqslant 1$ coefficients of the polynomial $S_{i,j}$.

Lemma 9.4. *A nonzero vector of $\mathbb{K}^{N\times 1}$ is in the right nullspace of $\mathbf{H}$ if and only if it is the coefficient vector of a solution $\mathbf{p} = [p_0, \ldots, p_{m-1}]$ to Problem 10.*

Proof. It is sufficient to consider a column vector $\mathbf{p} = [p_0, \ldots, p_{m-1}]$ that satisfies *(b)*. Then, looking at the high-degree terms in the identities in Eq. (9.2), we see that the condition *(c)* is equivalent to the following homogeneous system of linear equations over $\mathbb{K}$: for all i in $\{0, \ldots, n-1\}$ and all δ in $\{\delta_i - D_i, \ldots, \delta_i - 1\}$,

$$\sum_{0\leqslant j<m} \sum_{0\leqslant k<N_j} S_{i,j}^{(\delta-k)} p_j^{(N_j-1-k)} = 0.$$

The matrix obtained by considering all these equations is precisely the matrix $\mathbf{H}$. ☙

We will use the approach recalled in Section 9.1 to find a nonzero nullspace element for $\mathbf{H}$, with respect to the displacement operator $\Delta'_{D,N}$. To make sure that this approach is efficient, we need to prove that the displacement rank of $\mathbf{H}$ with respect to $\Delta'_{D,N}$ is bounded by a value α which is not too large. In addition, we also have to efficiently compute a generator of length α for $\mathbf{H}$, that is, a pair of matrices $(\mathbf{V}, \mathbf{W}) \in \mathbb{K}^{D\times\alpha} \times \mathbb{K}^{\alpha\times N}$ such that $\mathbf{H} - \mathbf{Z}_D\mathbf{H}\mathbf{Z}_N = \mathbf{V}\mathbf{W}$. We will see that here, computing such a generator boils down to computing the coefficients of the polynomials $S_{i,j}$. The cost incurred by computing this generator is summarized in the following lemma; combined with Proposition 9.1 and Lemma 9.4, this proves Theorem 2.25.

Lemma 9.5. *The displacement rank of $\mathbf{H}$ with respect to $\Delta'_{D,N}$ is at most $m + n$. Furthermore, one can compute a corresponding generator of length $m + n$ for $\mathbf{H}$ using $\mathcal{O}((m + n)\mathsf{M}(D))$ operations in $\mathbb{K}$.*

Proof. We are going to exhibit two matrices $\mathbf{V} \in \mathbb{K}^{D\times(m+n)}$ and $\mathbf{W} \in \mathbb{K}^{(m+n)\times N}$ such that $\mathbf{H} - \mathbf{Z}_D\mathbf{H}\mathbf{Z}_N = \mathbf{V}\mathbf{W}$. Because of the structure of $\mathbf{H}$, at most n rows and m columns of the matrix $\mathbf{H} - \mathbf{Z}_D\mathbf{H}\mathbf{Z}_N$ are nonzero. More precisely, only the first row and the last column of each $D_i \times N_j$ block of this matrix can be nonzero. Indexing the rows (resp. the columns) of $\mathbf{H} - \mathbf{Z}_D\mathbf{H}\mathbf{Z}_N$ from 0 to $D - 1$ (resp. from 0 to $N - 1$), only the n rows with indices of the form $r_i = D_0 + \cdots + D_{i-1}$ for $i = 0, \ldots, n-1$ can be nonzero, and only the

m columns with indices of the form $c_j = N_0 + \cdots + N_j - 1$ for $j = 0, \ldots, m-1$ can be nonzero.

For two integers i, K with $0 \leqslant i < K$, define $\mathbf{c}_{i,K} = [0 \ \cdots \ 0 \ 1 \ 0 \ \cdots \ 0]^\mathsf{T} \in \mathbb{K}^{K\times 1}$ the coordinate vector with 1 at position i, and

$$\mathbf{c}^{(\mathbf{V})} = [\mathbf{c}_{r_i,D}]_{0\leqslant i<n} \in \mathbb{K}^{D\times n},$$
$$\mathbf{c}^{(\mathbf{W})} = \left[\mathbf{c}_{c_j,N}\right]^\mathsf{T}_{0\leqslant j<m} \in \mathbb{K}^{m\times N}.$$

For given i in $\{0, \ldots, n-1\}$ and j in $\{0, \ldots, m-1\}$, we will consider $\mathbf{v}_{i,j} = [v_{i,j}^{(k)}]_{0\leqslant k<D_i}$ in $\mathbb{K}^{D_i\times 1}$ and $\mathbf{w}_{i,j} = [w_{i,j}^{(k)}]_{0\leqslant k<N_j}$ in $\mathbb{K}^{1\times N_j}$, which are respectively the last column and the first row of the block (i,j) in $\mathbf{H} - \mathbf{Z}_D\mathbf{H}\mathbf{Z}_N$, up to a minor point: the first entry of $v_{i,j}$ is set to zero. The coefficients $v_{i,j}^{(k)}$ and $w_{i,j}^{(k)}$ can then be expressed in terms of the entries $\mathbf{H}_{i,j}^{(u,v)} = S_{i,j}^{(u+v+\gamma_j)}$ of the Hankel matrix $\mathbf{H}_{i,j} = [\mathbf{H}_{i,j}^{(u,v)}]_{0\leqslant u<D_i, 0\leqslant v<N_j}$ as follows:

$$v_{i,j}^{(k)} = \begin{cases} 0 & \text{if } k = 0, \\ \mathbf{H}_{i,j}^{(k,N_j-1)} - \mathbf{H}_{(i,j+1)}^{(k-1,0)} & \text{if } 1 \leqslant k < D_i, \end{cases} \tag{9.5}$$

$$w_{i,j}^{(k)} = \begin{cases} \mathbf{H}_{i,j}^{(0,k)} - \mathbf{H}_{i-1,j}^{(D_{i-1}-1,k+1)} & \text{if } k < N_j - 1, \\ \mathbf{H}_{i,j}^{(0,N_j-1)} - \mathbf{H}_{i-1,j+1}^{(D_{i-1}-1,0)} & \text{if } k = N_j - 1. \end{cases} \tag{9.6}$$

Note that here, we use the convention that an indexed object is zero when the index is out of the allowed bounds for this object.

Then, we define $\mathbf{V}_j$ and $\mathbf{W}_i$ as

$$\mathbf{V}_j = \begin{bmatrix} \mathbf{v}_{0,j} \\ \vdots \\ \mathbf{v}_{n-1,j} \end{bmatrix} \in \mathbb{K}^{D\times 1} \quad \text{and}$$
$$\mathbf{W}_i = [\mathbf{w}_{i,0} \ \cdots \ \mathbf{w}_{i,m-1}] \in \mathbb{K}^{1\times N},$$

and we define $\mathbf{V}'$ and $\mathbf{W}'$ as

$$\mathbf{V}' = \begin{bmatrix} \mathbf{V}_0 & \cdots & \mathbf{V}_{m-1} \end{bmatrix} \in \mathbb{K}^{D\times m} \quad \text{and}$$
$$\mathbf{W}' = \begin{bmatrix} \mathbf{W}_0 \\ \vdots \\ \mathbf{W}_{n-1} \end{bmatrix} \in \mathbb{K}^{n\times N}.$$

Now, one can easily verify that the matrices

$$\mathbf{V} = \begin{bmatrix} \mathbf{V}' & \mathbf{c}^{(\mathbf{V})} \end{bmatrix} \in \mathbb{K}^{D\times(m+n)} \tag{9.7a}$$

and

$$\mathbf{W} = \begin{bmatrix} \mathbf{c}^{(\mathbf{W})} \\ \mathbf{W}' \end{bmatrix} \in \mathbb{K}^{(m+n)\times N} \tag{9.7b}$$

are generators for $\mathbf{H}$, that is, $\mathbf{H} - \mathbf{Z}_M \mathbf{H} \mathbf{Z}_N = \mathbf{V}\mathbf{W}$.

We notice that all we need in order to compute the generators $\mathbf{V}$ and $\mathbf{W}$ are the last $D_i + N_j - 1$ coefficients of $S_{i,j}(X) = S_{i,j}^{(0)} + S_{i,j}^{(1)} X + \cdots + S_{i,j}^{(\delta_i - 1)} X^{\delta_i - 1}$ for $0 \leqslant i < n$ and $0 \leqslant j < m$. Now, recall that

$$S_{i,j} = \frac{X^{\gamma_j} \overline{F_{i,j}}}{\overline{\mathfrak{m}_i}} \bmod X^{\delta_i} = \frac{X^{\delta_i - (D_i + N_j - 1)} \overline{F_{i,j}}}{\overline{\mathfrak{m}_i}} \bmod X^{\delta_i}.$$

Thus, the first $\delta_i - (D_i + N_j - 1)$ coefficients of $S_{i,j}$ are zero, and the last $D_i + N_j - 1$ coefficients of $S_{i,j}$ are the coefficients of

$$S_{i,j}^{\star} = \frac{\overline{F_{i,j}}}{\overline{\mathfrak{m}_i}} \bmod X^{D_i + N_j - 1}, \tag{9.8}$$

which can be computed in $\mathcal{O}(\mathsf{M}(D_i + N_j))$ operations in $\mathbb{K}$ by fast power series division. By expanding products, we see that $\mathsf{M}(D_i + N_j) = \mathcal{O}(\mathsf{M}(D_i) + \mathsf{M}(N_j))$. Summing the costs, we obtain an upper bound of the form

$$\mathcal{O}\left(\sum_{0 \leqslant i < n} \sum_{0 \leqslant j < m} \mathsf{M}(D_i) + \mathsf{M}(N_j) \right),$$

which is in $\mathcal{O}((m\mathsf{M}(D) + n\mathsf{M}(N))$ using the super-linearity of $\mathsf{M}(\cdot)$. Since we assumed in Eq. (9.1) that $N \leqslant D + 1$, this is in $\mathcal{O}((n + m)\mathsf{M}(D))$. ☙

9.3 Directly computing a solution via a Toeplitz-like system

In this section, we propose an alternative approach to Problem 10, outlined in Algorithm 20 below, which leads to the same asymptotic running time as the algorithm of the previous section, but follows a more direct solution path by avoiding the derivation of extended key equations as in Lemma 9.3. As recalled above, our input consists of the monic moduli $\mathfrak{M}$ and the matrix $\mathbf{F}$, and we look for a column vector $\mathbf{p}$ such that the following conditions hold: *(a)* $\mathbf{p}$ is nonzero, *(b)* $\operatorname{cdeg}_{\mathbf{N}}(\mathbf{p}) < 0$, and *(c)* $\mathbf{F}\mathbf{p} = 0 \bmod \mathfrak{M}$.

In addition, for $k \geqslant 0$, we denote by $F_{i,j}^{(k)}$ and $\mathfrak{m}_i^{(k)}$ the coefficients of degree k of $F_{i,j}$ and $\mathfrak{m}_i$, respectively, and we denote by $\mathbf{C}_i$ the companion matrix of $\mathfrak{m}_i$; explicitly,

$$\mathbf{C}_i = \begin{bmatrix} 0 & 0 & \cdots & 0 & -\mathfrak{m}_i^{(0)} \\ 1 & 0 & \cdots & 0 & -\mathfrak{m}_i^{(1)} \\ 0 & 1 & \cdots & 0 & -\mathfrak{m}_i^{(2)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -\mathfrak{m}_i^{(D_i - 1)} \end{bmatrix} \in \mathbb{K}^{D_i \times D_i}.$$

In particular, if p is a polynomial of degree less than D_i with coefficient vector $\mathbf{v} \in \mathbb{K}^{D_i \times 1}$, then the product $\mathbf{C}_i \mathbf{v} \in \mathbb{K}^{D_i \times 1}$ is the coefficient vector of the polynomial $Xp \bmod \mathfrak{m}_i$.

Algorithm 19 – SolVecViaHankel
(Small solution vector via a mosaic-Hankel system)

Input:

- positive integers n, m, $D_0, \ldots, D_{n-1}$, $N_0, \ldots, N_{m-1}$,
- polynomial tuples $\{(\mathfrak{m}_i, F_{i,0}, \ldots, F_{i,m-1})\}_{0 \leqslant i < n}$ in $\mathbb{K}[X]^{m+1}$ such that for all i, $\mathfrak{m}_i$ is monic of degree D_i and $\deg(F_{i,j}) < D_i$ for all j.

Output: polynomials $p_0, \ldots, p_{m-1}$ in $\mathbb{K}[X]$ such that *(a)*, *(b)*, *(c)*.

1. Compute the coefficients $S_{i,j}^{(\gamma_j + r)}$ for $r < D_i + N_j - 1$, $i < n$, and $j < m$; that is, the coefficients of $S_{i,j}^{\star}$ as defined in Eq. (9.8)
2. $\mathbf{v}_{i,j}$ and $\mathbf{w}_{i,j}$, for $i < n$ and $j < m \leftarrow$ as defined in Eq. (9.5) and Eq. (9.6)
3. $r_i \leftarrow D_0 + \cdots + D_{i-1}$ for $i < n$; $c_j \leftarrow N_0 + \cdots + N_j - 1$ for $j < m$
4. $\mathbf{V}$ and $\mathbf{W} \leftarrow$ generators as defined in Eq. (9.7)
5. Run the algorithm of Proposition 9.1 with input $\mathbf{V}$ and $\mathbf{W}$; if there is no solution then exit with no solution, otherwise retrieve the coefficients of the ouput as polynomials $p_0, \ldots, p_{m-1}$
6. Return $\mathbf{p} = [p_j]_j \in \mathbb{K}[X]^{m \times 1}$

We are going to see that solving Problem 10 is equivalent to finding a nonzero solution to a homogeneous linear system whose matrix is $\mathbf{T} = [\mathbf{T}_{i,j}]_{0 \leqslant i < n, 0 \leqslant j < m} \in \mathbb{K}^{D \times N}$, where for $i < n$ and $j < m$, $\mathbf{T}_{i,j} \in \mathbb{K}^{D_i \times N_j}$ is a matrix which depends on the coefficients of $F_{i,j}$ and $\mathfrak{m}_i$. We make the same assumption as in the previous section: $N \leqslant D + 1$ holds, without loss of generality.

For i, j as above and for $k \in \mathbb{Z}_{\geqslant 0}$, let $\alpha_{i,j}^{(k)} \in \mathbb{K}^{D_i \times 1}$ be the coefficient vector of the polynomial $X^k F_{i,j} \bmod \mathfrak{m}_i$, so that these vectors are given by

$$\alpha_{i,j}^{(0)} = \begin{bmatrix} F_{i,j}^{(0)} \\ \vdots \\ F_{i,j}^{(D_i - 1)} \end{bmatrix} \quad \text{and} \quad \alpha_{i,j}^{(k+1)} = \mathbf{C}_i \alpha_{i,j}^{(k)}.$$

Let then $\mathbf{T} = [\mathbf{T}_{i,j}]_{0 \leqslant i < n, 0 \leqslant j < m} \in \mathbb{K}^{D \times N}(\mathbf{T}_{i,j})$, where for every $i < n$ and $j < m$, the block $\mathbf{T}_{i,j} \in \mathbb{K}^{D_i \times N_j}$ is defined by

$$\mathbf{T}_{i,j} = \begin{bmatrix} \alpha_{i,j}^{(0)} & \cdots & \alpha_{i,j}^{(N_j - 1)} \end{bmatrix}.$$

Lemma 9.6. *A nonzero vector of $\mathbb{K}^{N \times 1}$ is in the right nullspace of $\mathbf{T}$ if and only if it is the coefficient vector of a solution $\mathbf{p} = [p_j]_j$ to Problem 10.*

Proof. By definition $\mathbf{T}_{i,j}$ is the $D_i \times N_j$ matrix of the mapping $p \mapsto F_{i,j} p \bmod \mathfrak{m}_i$, for p in $\mathbb{K}[X]$ of degree less than N_j. Thus, if $\mathbf{p}$ satisfies the degree constraint in the condition

(b), by applying $\mathbf{T}$ to the coefficient vector of this $\mathbf{p}$ we obtain the coefficients of the remainders $\sum_{0\leqslant j<m} F_{i,j}p_j \bmod \mathfrak{m}_i$ for $i = 0, \ldots, n-1$. The claimed equivalence follows immediately. ☙

The following lemma shows that $\mathbf{T}$ possesses a Toeplitz-like structure, with displacement rank at most $m+n$.

Lemma 9.7. *The displacement rank of* $\mathbf{T}$ *with respect to* $\Delta_{D,N}$ *is at most* $m+n$. *Furthermore, one can compute a corresponding generator of length* $m+n$ *for* $\mathbf{T}$ *using* $\mathcal{O}((m+n)\mathsf{M}(D))$ *operations in* $\mathbb{K}$.

Proof. We begin by giving two matrices $\mathbf{P} \in \mathbb{K}^{D\times(n+m)}$ and $\mathbf{Q} \in \mathbb{K}^{(n+m)\times N}$ such that $\Delta_{D,N}(\mathbf{T})$ is equal to the product $\mathbf{PQ}$. Define first the matrix

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_0 & 0 & \cdots & 0 \\ 0 & \mathbf{C}_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{C}_{n-1} \end{bmatrix} \in \mathbb{K}^{D\times D}.$$

Up to n columns, $\mathbf{C}$ coincides with $\mathbf{Z}_D$; we make this explicit as follows. For $0 \leqslant i < n$, we define

$$\mathbf{v}_i = \begin{bmatrix} \mathfrak{m}_i^{(0)} \\ \vdots \\ \mathfrak{m}_i^{(D_i-1)} \end{bmatrix} \in \mathbb{K}^{D_i\times 1}, \tag{9.9a}$$

$$\mathbf{V}_i = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \mathbf{v}_i \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \in \mathbb{K}^{D\times 1}, \qquad \mathbf{W}_i = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \in \mathbb{K}^{D\times D}, \tag{9.9b}$$

where the last entry of $\mathbf{v}_i$ in $\mathbf{V}_i$ and the coefficient 1 in $\mathbf{W}_i$ have the same index, namely $D_0 + \cdots + D_i - 1$. (Hence the last vector $\mathbf{V}_{n-1}$ only contains $\mathbf{V}_{n-1}$, without a 1 after it.) Then, defining $\mathbf{V} = [\mathbf{V}_0 \ \cdots \ \mathbf{V}_{n-1}] \in \mathbb{K}^{D\times n}$ and $\mathbf{W} = [\mathbf{W}_0 \ \cdots \ \mathbf{W}_{n-1}] \in \mathbb{K}^{D\times n}$, we obtain

$$\mathbf{C} = \mathbf{Z}_D - \mathbf{V}_0\mathbf{W}_0^\mathsf{T} - \cdots - \mathbf{V}_{n-1}\mathbf{W}_{n-1}^\mathsf{T} = \mathbf{Z}_D - \mathbf{V}\mathbf{W}^\mathsf{T}.$$

As before, we use the convention that an indexed object is zero when the index is out of the allowed bounds for this object. For $0 \leqslant j < m$, let us further define

$$\mathbf{V}'_j = \begin{bmatrix} \alpha_{0,j}^{(0)} \\ \vdots \\ \alpha_{n-1,j}^{(0)} \end{bmatrix} - \begin{bmatrix} \alpha_{0,j-1}^{(N_{j-1})} \\ \vdots \\ \alpha_{n-1,j-1}^{(N_{j-1})} \end{bmatrix} \in \mathbb{K}^{D\times 1} \tag{9.10a}$$

and

$$\mathbf{W}'_j = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \in \mathbb{K}^{N\times 1}, \tag{9.10b}$$

with the coefficient 1 in $\mathbf{W}'_j$ at index $N_0 + \cdots + N_{j-1}$, and the compound matrices

$$\begin{aligned} \mathbf{V}' &= \begin{bmatrix} \mathbf{V}'_0 & \cdots & \mathbf{V}'_{m-1} \end{bmatrix} \in \mathbb{K}^{D\times m}, \\ \mathbf{W}' &= \begin{bmatrix} \mathbf{W}'_0 & \cdots & \mathbf{W}'_{m-1} \end{bmatrix} \in \mathbb{K}^{N\times m}. \end{aligned}$$

Then, we claim that the matrices

$$\mathbf{M} = \begin{bmatrix} -\mathbf{V} & \mathbf{V}' \end{bmatrix} \in \mathbb{K}^{D\times(m+n)} \tag{9.11a}$$

and

$$\mathbf{N} = \begin{bmatrix} \mathbf{W}^\mathsf{T}\mathbf{T}\mathbf{Z}_N^\mathsf{T} \\ \mathbf{W}'^\mathsf{T} \end{bmatrix} \in \mathbb{K}^{(m+n)\times N} \tag{9.11b}$$

are generators for $\mathbf{T}$ for the Toeplitz-like displacement structure, that is,

$$\mathbf{T} - \mathbf{Z}_D\mathbf{T}\mathbf{Z}_N^\mathsf{T} = \mathbf{M}\mathbf{N}.$$

By construction, we have $\mathbf{C}\mathbf{T} = (\mathbf{P}_{i,j})_{i<n,j<m} \in \mathbb{K}^{D\times N}$, with the block $\mathbf{P}_{i,j}$ being given by

$$\mathbf{P}_{i,j} = \mathbf{C}_i\mathbf{T}_{i,j} = \begin{bmatrix} \alpha_{i,j}^{(1)} & \cdots & \alpha_{i,j}^{(N_j-1)} & \alpha_{i,j}^{(N_j)} \end{bmatrix} \in \mathbb{K}^{D_i\times N_j}.$$

As a consequence, $\mathbf{T} - \mathbf{C}\mathbf{T}\mathbf{Z}_N^\mathsf{T} = \mathbf{V}'\mathbf{W}'^\mathsf{T}$, so finally we get, as claimed,

$$\begin{aligned} \mathbf{T} - \mathbf{Z}_D\mathbf{T}\mathbf{Z}_N^\mathsf{T} &= \mathbf{T} - (\mathbf{C} + \mathbf{V}\mathbf{W}^\mathsf{T})\mathbf{T}\mathbf{Z}_N^\mathsf{T} \\ &= \mathbf{T} - \mathbf{C}\mathbf{T}\mathbf{Z}_N^\mathsf{T} - \mathbf{V}\mathbf{W}^\mathsf{T}\mathbf{T}\mathbf{Z}_N^\mathsf{T} \\ &= \mathbf{V}'\mathbf{W}'^\mathsf{T} - \mathbf{V}\mathbf{W}^\mathsf{T}\mathbf{A}\mathbf{Z}_N^\mathsf{T} \\ &= \mathbf{M}\mathbf{N}. \end{aligned}$$

To compute $\mathbf{Y}$ and $\mathbf{Z}$, the only non-trivial steps are those giving $\mathbf{V}'$ and $\mathbf{W}^\mathsf{T}\mathbf{T}$. For the former, we have to compute the coefficients of $X^{N_j}F_{i,j} \bmod \mathfrak{m}_i$ for every $i < n$ and $j < m-1$. For fixed i and j, this can be done using fast Euclidean division in $\mathcal{O}(\mathsf{M}(D_i + N_j))$ operations in $\mathbb{K}$, which is in $\mathcal{O}(\mathsf{M}(D_i) + \mathsf{M}(N_j))$. Summing over the indices $i < n$ and $j < m-1$, this gives a total cost of $\mathcal{O}(m\mathsf{M}(D) + n\mathsf{M}(N))$ operations. This is in $\mathcal{O}((n+m)\mathsf{M}(D))$, since by assumption $N \leqslant D+1$.

Finally, we show that $\mathbf{W}^\mathsf{T}\mathbf{T}$ can be computed using $\mathcal{O}((n+m)\mathsf{M}(D))$ operations as well. Computing this matrix amounts to computing the rows of $\mathbf{T}$ of indices $D_0 + \cdots + D_i - 1$, for $i < n$. By construction of $\mathbf{T}$, this means that we want to compute the coefficients of

degree $D_i - 1$ of $X^k F_{i,j} \bmod \mathfrak{m}_i$ for $k = 0, \ldots, N_j - 1$ and for all i, j. Unfortunately, the naive approach leads to a cost proportional to DN operations, which is not acceptable. However, for i and j fixed, Lemma 9.8 below shows how to do this computation using only $\mathcal{O}(\mathsf{M}(D_i) + \mathsf{M}(N_j))$ operations, which leads to the announced cost by summing over i and j. ☙

Together with Proposition 9.1 and Lemma 9.6, this result completes our second proof of Theorem 2.25, and leads to following Algorithm 20.

Algorithm 20 – SolVecViaToeplitz
(Small solution vector via a Toeplitz-like system)

Input:

- positive integers n, m, $D_0, \ldots, D_{n-1}$, $N_0, \ldots, N_{m-1}$,
- polynomial tuples $\{(\mathfrak{m}_i, F_{i,0}, \ldots, F_{i,m-1})\}_{0 \leqslant i < n}$ in $\mathbb{K}[X]^{m+1}$ such that for all i, $\mathfrak{m}_i$ is monic of degree D_i and $\deg(F_{i,j}) < D_i$ for all j.

Output: polynomials $p_0, \ldots, p_{m-1}$ in $\mathbb{K}[X]$ such that *(a)*, *(b)*, *(c)*.

1. $\mathbf{v}_i$ and $\mathbf{V}_i$, for $i < n \leftarrow$ as defined in Eq. (9.9); $\mathbf{V} \leftarrow [\mathbf{V}_0 \ \cdots \ \mathbf{V}_{n-1}]$
2. $\mathbf{W}'_j$, for $j < m \leftarrow$ as defined in Eq. (9.10); $\mathbf{W}' \leftarrow [\mathbf{W}'_0 \ \cdots \ \mathbf{W}'_{m-1}]$
3. $\alpha_{i,j}^{(N_j)}$, for $i < n, j < m - 1 \leftarrow$ the coefficients of $X^{N_j} F_{i,j} \bmod \mathfrak{m}_i$
4. $\mathbf{V}'_j$, for $j < n \leftarrow$ as defined in Eq. (9.10); $\mathbf{V}' \leftarrow [\mathbf{V}'_0 \ \cdots \ \mathbf{V}'_{m-1}]$
5. $\mathbf{W}^\mathsf{T}\mathbf{T} \leftarrow$ for each $i < n$, its row of index i is the row of index $D_0 + \cdots + D_i - 1$ of $\mathbf{T}$, that is, the coefficients of degree $D_i - 1$ of $X^k F_{i,j} \bmod \mathfrak{m}_i$ for $k < N_j$ and $j < m$ (see Lemma 9.8 for fast computation)
6. $\mathbf{M}$ and $\mathbf{N} \leftarrow$ generators as defined in Eq. (9.11)
7. Run the algorithm of Proposition 9.1 with input $\mathbf{M}$ and $\mathbf{N}$
   $\rightsquigarrow$ if there is no solution then exit with no solution, otherwise retrieve the coefficients of the output as polynomials $p_0, \ldots, p_{m-1}$
8. Return $\mathbf{p} = [p_j]_j \in \mathbb{K}[X]^{m \times 1}$

Lemma 9.8. *Let $\mathfrak{m} \in \mathbb{K}[X]$ be monic of degree D, let $F \in \mathbb{K}[X]$ be of degree less than D, and for $i \geqslant 0$ let c_i denote the coefficient of degree $D - 1$ of $X^i F \bmod \mathfrak{m}$. Then, for $d \geqslant 1$ one can compute $c_0, \ldots, c_{d-1}$ using $\mathcal{O}(\mathsf{M}(D) + \mathsf{M}(d))$ operations in $\mathbb{K}$.*

Proof. Writing $F = \sum_{0 \leqslant j < D} f_j X^j$ we have $X^i F \bmod \mathfrak{m} = \sum_{0 \leqslant j < D} f_j (X^{i+j} \bmod \mathfrak{m})$. Hence $c_i = \sum_{0 \leqslant j < D} f_j b_{i+j}$, with b_i denoting the coefficient of degree $D - 1$ of $X^i \bmod \mathfrak{m}$. Since $b_0 = \cdots = b_{D-2} = 0$ and $b_{D-1} = 1$, we can deduce $c_0, \ldots, c_{d-1}$ from $b_{D-1}, b_D, \ldots, b_{D+d-2}$ in time $\mathcal{O}(\mathsf{M}(d))$ by multiplication by the lower triangular Toeplitz matrix $[f_{D+j-i-1}]_{i,j}$ of order $d - 1$.

Thus, we are left with the question of computing the $d - 1$ coefficients $b_D, \ldots, b_{D+d-2}$. Writing $\mathfrak{m}$ as $\mathfrak{m} = X^D + \sum_{0 \leqslant j < D} p_j X^j$ and using the fact that $X^i \mathfrak{m} \bmod \mathfrak{m} = 0$ for all

$i \geqslant 0$, we see that the b_i are generated by a linear recurrence of order D with constant coefficients:

$$b_{i+D} + \sum_{0 \leqslant j < D} p_j b_{i+j} = 0 \quad \text{for all } i \geqslant 0.$$

Consequently, $b_D, \ldots, b_{D+d-2}$ can be deduced from $b_0, \ldots, b_{D-1}$ in time $\mathcal{O}(\frac{d}{D}\mathsf{M}(D))$, which is in $\mathcal{O}(\mathsf{M}(D) + \mathsf{M}(d))$, by $\lceil \frac{d-1}{D} \rceil$ calls to Shoup's algorithm for extending a linearly recurrent sequence [Sho91, Theorem 3.1]. ☙

10

Coppersmith technique over the univariate polynomials

In this chapter, we reduce the interpolation step of the Coppersmith technique over the univariate polynomials (Problem 14) to a system of linear modular equation (Problem 9). Following the original Coppersmith technique over $\mathbb{Z}$ [Cop96], we first present the solution based on row reduction, from [CH15]; it uses $\mathcal{O}\tilde{}(\lambda^\omega \mu\nu)$ field operations. Our reduction to Problem 9, for which we gave fast algorithms in both preceding Chapters 8 and 9, leads to a faster solution. Precisely, we will prove the following result, which uses notation from Section 3.1.5.

Proposition 10.1. *Problem 14 can be solved using $\mathcal{O}\tilde{}(\lambda^{\omega-1}\mu^2\nu d)$ operations in $\mathbb{K}$.*

In both the approaches of [CH15] and ours, the list-size parameter λ is used to identify the sought $Q \in \mathbb{K}[X][Y]$ of Y-degree at most λ with the vector $[Q_0, Q_1, \ldots, Q_\lambda]$ such that $Q = Q_0 + Q_1 Y + \cdots + Q_\lambda Y^\lambda$. This is a linearization of the problem with respect to the variable Y: instead of considering bivariate polynomials, we will work with $\mathbb{K}[X]$-submodules of $\mathbb{K}[X]^{\lambda+1}$, allowing us to benefit from the tools available for these submodules.

10.1 The approach based on row reduction

The strategy here is to first build a known basis of the module of all $[Q_0 \ \cdots \ Q_\lambda]$ such that Q belongs to the ideal $\mathcal{I}$. Then, the degree constraint on Q will be satisfied through the computation of a shifted reduced form of this basis, with the shift being induced by the weight w in Problem 14.

Building the basis. This construction, detailed in [CH11], is analogous to a construction encountered in Howgrave-Graham's version of the Coppersmith technique over the integers [HG01]. The same construction was used in algorithms based on row reduction for the Guruswami-Sudan algorithm [Ale02, Rei03, LO08, BB10, Ber11].

The condition that Q belongs to the ideal $\mathcal{I} = \langle F, M \rangle^\mu$, with the degree constraint $\deg_Y(Q) \leqslant \lambda$, is equivalent to having Q in the $\mathbb{K}[X]$-module generated by the basis

$$\mathcal{E} \ = \ \{M^{\mu-i} Y^k F^i, \ i < \mu, k < d, id + k \leqslant \lambda\} \ \cup \ \{Y^k F^\mu, \ k \leqslant \lambda - \mu d\}. \tag{10.1}$$

This basis is represented as a matrix $\mathbf{B} \in \mathbb{K}[X]^{(\lambda+1)\times(\lambda+1)}$, which is shown in Fig. 10.1. We observe that $\mathbf{B}$ inherits some structure from the bivariate nature of the original problem: this is however not exploited by the fastest known algorithms.

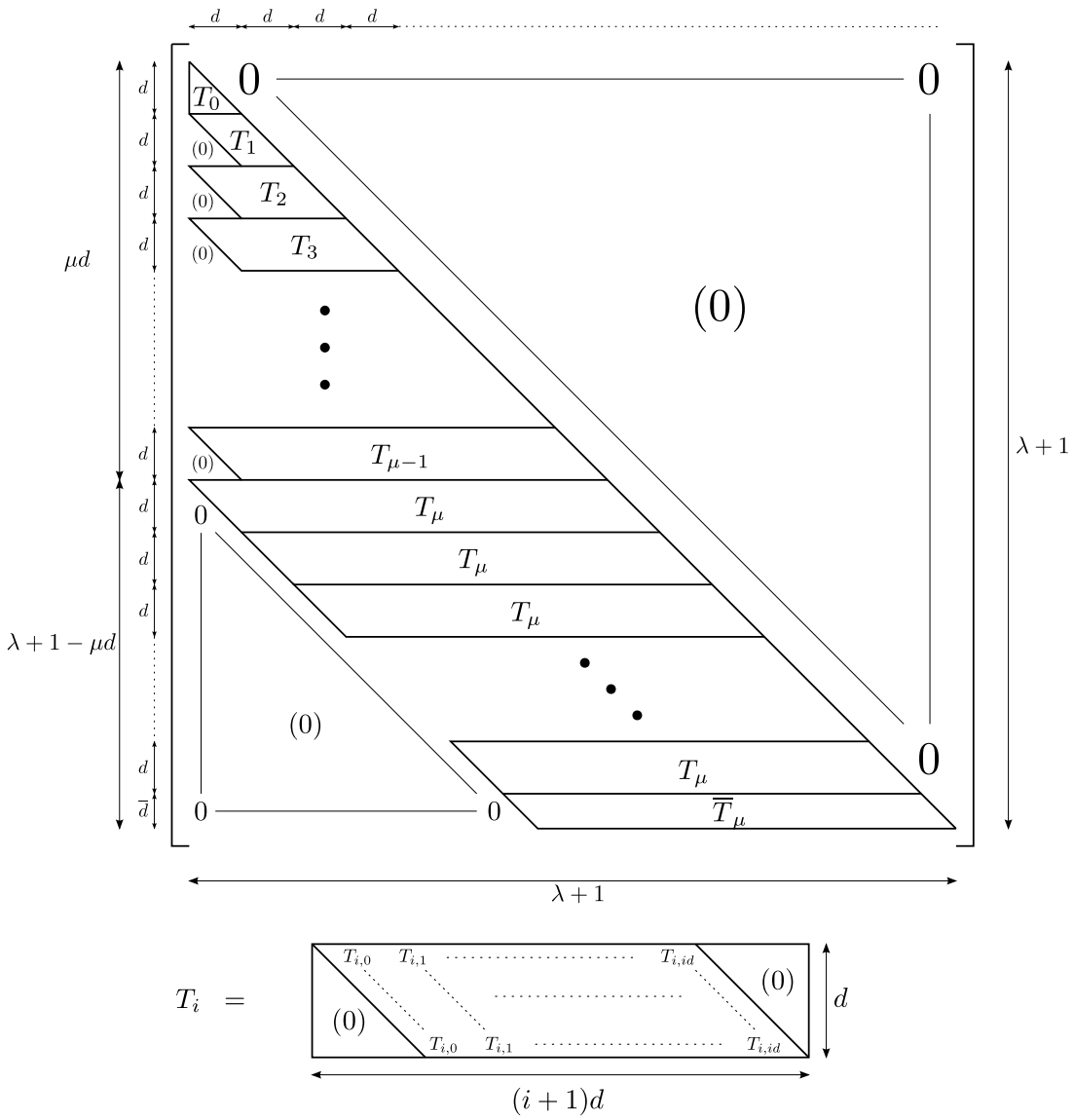

Figure 10.1: The basis $\mathbf{B}$ of the $\mathbb{K}[X]$-module, represented as a square $(\lambda+1)\times(\lambda+1)$ polynomial matrix. Each T_i is a trapezoidal Toeplitz block with the shape given above, where $[T_{i,0},\ldots,T_{i,id}]$ is the vector of the coefficients in X of $M^{\mu-i}F(Y)^i$. The last block $\overline{T}_\mu$ is formed by the upper $\overline{d}$ rows of T_μ, where $\overline{d} = (\lambda+1) \text{ rem } d$.

This matrix $\mathbf{B}$ is lower triangular with diagonal entries

$$\underbrace{M^\mu,\ldots,M^\mu}_{d},\underbrace{M^{\mu-1},\ldots,M^{\mu-1}}_{d},\ldots,\underbrace{M,\ldots,M}_{d},\underbrace{1,\ldots,1}_{\lambda+1-\mu d}.$$

To obtain the polynomials in $\mathcal{E}$, it is enough to compute $M^\mu, M^{\mu-1}F,\ldots,MF^{\mu-1},F^\mu$; this can be done using a total of $\mathcal{O}\tilde{}(\mu^3\nu d)$ operations in $\mathbb{K}$.

The entries of $\mathbf{B}$ can be computed modulo M^μ (except for the top-left entry which is M^μ itself) without loss of generality. Thus, the maximum degree of the entries of $\mathbf{B}$ is $\mu\nu$, and the dense representation of $\mathbf{B}$ uses $\mathcal{O}(\lambda\mu^2\nu d)$ elements from $\mathbb{K}$, a bound that is reached generically.

Note that, in this bound, we use the fact that on each row of $\mathbf{B}$, there are at most $\mu\nu$ nonzero entries. Yet, currently known fast reduction algorithms will ignore this banded shape of $\mathbf{B}$: it is considered as an $(\lambda+1)\times(\lambda+1)$ matrix of degree $\mu\nu$. Such a matrix has size $\mathcal{O}(\lambda^2\mu\nu)$, which is beyond our target cost $\mathcal{O}\tilde{}(\lambda^{\omega-1}\mu^2\nu d)$.

Reducing the basis. To solve Problem 14, we will then compute a shifted reduced form of $\mathbf{B}$. In order to reflect the constraint $\deg_X(Q(X^wY)) < \mu t$, we choose the shift $\mathbf{s} = (0, w, \ldots, \lambda w)$. Since the fastest known $\mathbf{0}$-reduction algorithms are designed for the uniform shift [GJV03, GSSV12], we take $\mathbf{s}$ into account by computing a $\mathbf{0}$-reduced form of $\mathbf{BX^s}$. We have seen that the largest degree of the entries of $\mathbf{B}$ is $\mu\nu = \deg(M^\mu)$. Multiplying the columns of $\mathbf{B}$ by the powers $1, X^w, \ldots, X^{\lambda w}$ does not impact this bound: since we have $\lambda w \leqslant \mu t \leqslant \mu\nu$ by choice of the parameters, then the degree of the new matrix $\mathbf{BX^s}$ remains in $\mathcal{O}(\mu\nu)$.

Then, using one of the algorithms in [GJV03, GSSV12], we find a $\mathbf{0}$-reduced form of $\mathbf{BX^s}$ using $\mathcal{O}\tilde{}(\lambda^\omega\mu\nu)$ operations in $\mathbb{K}$. Since row reduction operates with left-unimodular operations, this reduced form still has $\mathbf{X^s}$ as a right factor, and can be written $\mathbf{PX^s}$. Having the latter matrix $\mathbf{0}$-reduced means that $\mathbf{P}$ is $\mathbf{s}$-reduced, and therefore $\mathbf{P}$ is a $\mathbf{s}$-reduced form of $\mathbf{B}$. Then, among the rows of $\mathbf{P}$, it is guaranteed that there is one with minimal $\mathbf{s}$-degree. Writing it $[Q_0 \ \cdots \ Q_\lambda]$, it corresponds to $Q = Q_0 + Q_1Y + \cdots + Q_\lambda Y^\lambda$ that solves Problem 14 (unless no solution exists, in which case either $t^2 \leqslant w\nu d$ or the parameters μ and λ have not been chosen properly).

Besides, it is known that such a row with minimal degree satisfies

$$\max_{0\leqslant j\leqslant\lambda} \deg(Q_jX^{jw}) \leqslant \frac{\deg(\det(\mathbf{BX^s}))}{\lambda+1},$$

that is,

$$\deg(Q(X^wY)) \leqslant \frac{\frac{1}{2}\lambda(\lambda+1)w + \frac{1}{2}\mu(\mu+1)\nu d}{\lambda+1}.$$

Thus, to ensure the existence of a solution, we require that

$$\mu t > \frac{\frac{1}{2}\lambda(\lambda+1)w + \frac{1}{2}\mu(\mu+1)\nu d}{\lambda+1},$$

which is equivalent to Eq. (3.5).

Remarks. Here, the problem is not exactly the one of finding a reduced basis, but rather of finding a vector of sufficiently small degree in the module. First, the degree bound is so that we are close to looking for a vector of minimal degree in the module. Second, to the best of our knowledge, it is currently unknown whether and how one could compute a single vector of minimal degree faster than by computing a whole reduced basis, except in very specific situations.

Besides, one may rather want to build the basis which is in Hermite form, that is, with the shape $\left[\begin{smallmatrix}\mathbf{T} & \mathbf{0}\\ \mathbf{A} & \mathbf{I}\end{smallmatrix}\right]$ with $\mathbf{T}$ the principal $\mu d \times \mu d$ submatrix of $\mathbf{B}$. Then, since the first μd columns of this matrix have degree at most $\mu\nu$ and its remaining columns are constant, the average of its column degrees is $\mu^2\nu d/(\lambda+1)$. This fact may be exploited to obtain more efficiently the reduced basis, or a vector of sufficiently small degree, relying on some partial linearization techniques to make the column degrees more uniform. Such techniques, in the context of row reduction, are presented in Section 15.2.

We note in particular that our shifted Popov form algorithm in Section 3.2.2 and Chapter 15 computes the sought reduced form in $\tilde{\mathcal{O}}(\lambda^{\omega-1}\mu^2\nu d)$ operations, by taking this average column degree into account. However, it would start by the probabilistic computation of the Smith form of $\mathbf{B}$ and of a corresponding right-multiplier, in order to set up a system of linear modular equations. In the next section, we exploit the particularities of the considered module to circumvent the Smith form computation, and to rather directly construct the system of modular equations that describes the module.

10.2 Reducing to a system of linear modular equations

In this section, we present a reduction of the interpolation step Problem 14 to a system of linear modular equations Problem 9. Fast algorithms for such systems were presented in Section 2.5 and Chapter 8.

10.2.1 Introduction: the specific case $d = 1$

Here, we first present this reduction in a simple case, but nonetheless important: when $d = \deg(F) = 1$, which includes the Guruswami-Sudan algorithm. Although we follow a different presentation, to ease the transition to the general case in the next subsection, the material here is close to the derivation of extended key equations in [RR00, ZGA11, Zeh13] (see also Section 11.1).

In what follows, we write $F = Y - L$ for some $L \in \mathbb{K}[X]$ of degree less than ν. Assuming that $\mu = 1$, the reduction is as follows: we have $Q \in \mathcal{I} = \langle Y - L, M\rangle$ if and only if $Q(L) = 0 \bmod M$, which is rewritten as the single linear modular equation $Q_0 + Q_1 L + \cdots + Q_\lambda L^\lambda = 0 \bmod M$ (such an equation was used in [RR00] to speed-up the Sudan algorithm). Now, when $\mu \geqslant 1$, this was generalized in [ZGA11, Proposition 3] using the notion of *Hasse derivative*: roughly, $Q \in \mathcal{I} = \langle Y - L, M\rangle^\mu$ if and only if some type of derivatives of $Q(Y)$ at order $0, \ldots, \mu - 1$ vanish at L modulo $M^\mu, \ldots, M$, respectively. We refer to these references and to Section 11.1 for more details on this point of view.

Here, we study the latter property involving derivatives, reformulated as a change of basis which can be computed via some Taylor expansions. We are looking for $Q \in \mathbb{K}[X][Y]$ which belongs to the ideal $\mathcal{I}$ of $\mathbb{K}[X][Y]$ generated by

$$M^\mu, M^{\mu-1}(Y - L), \ldots, M(Y - L)^{\mu-1}, (Y - L)^\mu.$$

This is equivalent to

$$Q \;=\; \widehat{Q}_0 M^\mu + \widehat{Q}_1 M^{\mu-1}(Y - L) + \cdots + \widehat{Q}_{\mu-1} M(Y - L)^{\mu-1} + \widehat{Q}(Y)(Y - L)^\mu$$

for some $\widehat{Q}_0, \ldots, \widehat{Q}_{\mu-1} \in \mathbb{K}[X]$ and some $\widehat{Q} \in \mathbb{K}[X][Y]$. Adding the constraint $\deg(Q) \leqslant \lambda$, this is equivalent to Q belonging to the module $\mathcal{M} = \mathcal{I} \cap \mathbb{K}[X][Y]_{\leqslant\lambda}$ of dimension $\lambda + 1$ with basis

$$\mathcal{E} \;=\; \{M^\mu, M^{\mu-1}(Y-L), \ldots, M(Y-L)^{\mu-1}, (Y-L)^\mu, Y(Y-L)^\mu, \ldots, Y^{\lambda-\mu}(Y-L)^\mu\}.$$

In order to transform this property of belonging to $\mathcal{M}$ into a property of satisfying modular equations, one first rewrites $Q = Q_0 + \cdots + Q_\lambda Y^\lambda$ in the basis

$$\mathcal{E}' \;=\; \{1, Y-L, \ldots, (Y-L)^{\mu-1}, (Y-L)^\mu, Y(Y-L)^\mu, \ldots, Y^{\lambda-\mu}(Y-L)^\mu\}$$

of $\mathbb{K}[X][Y]_{\leqslant\lambda}$, that is,

$$Q = \widehat{Q}_0 + \widehat{Q}_1(Y-L) + \cdots + \widehat{Q}_{\mu-1}(Y-L)^{\mu-1} + \widehat{Q}(Y)(Y-L)^\mu$$

for some $\widehat{Q}_0, \ldots, \widehat{Q}_{\mu-1} \in \mathbb{K}[X]$ and some $\widehat{Q} \in \mathbb{K}[X][Y]$ of degree at most $\lambda - \mu$. We remark that the coefficients $\widehat{Q}_i$ of Q written in $\mathcal{E}$ are unique: ensuring that Q is in $\mathcal{M}$ is thus equivalent to ensuring the modular equations

$$M^{\mu-i} \;\text{ divides }\; \widehat{Q}_i \quad \text{for } \; i < \mu. \tag{10.2}$$

This is the linear system of modular equations over $\mathbb{K}[X]$ that we are going to focus on. It remains to show that we can compute the $\widehat{Q}_i$'s efficiently as $\mathbb{K}[X]$-linear combinations of the unknown Q_i's.

In the case here with $F = Y - L$, the computation of the $\widehat{Q}_i$'s is straightforward since explicit formula are known for the change of basis from $\{1, Y, \ldots, Y^\lambda\}$ to $\mathcal{E}'$, namely thanks to the Taylor formula

$$Q \;=\; \sum_{j \leqslant \lambda} Q_j (Y - L + L)^j \;=\; \sum_{i \leqslant \lambda} \left(\sum_{j \geqslant i} \binom{j}{i} Q_j L^{j-i} \right) (Y-L)^i.$$

This gives a formula for the coefficients $[\widehat{Q}_i]_i$ of Q in the basis $\mathcal{E}'$, as linear combinations with explicit coefficients in $\mathbb{K}[X]$ of the coefficients $[Q_j]_j$ in the basis $\mathcal{E}$; namely,

$$\widehat{Q}_i = \sum_{j \leqslant \lambda} f_{ij} Q_j \quad \text{with} \quad f_{ij} = \binom{j}{i} L^{j-i}.$$

We have $f_{ij} = 0$ for $i > j$ and $f_{jj} = 1$. From the divisibility conditions in Eq. (10.2), we thus obtain the system of linear modular equations

$$\sum_{j \leqslant \lambda} f_{ij} Q_j \;=\; 0 \bmod M^{\mu-i} \quad \text{for } \; i < \mu.$$

In particular, due to the nature of these equations, the coefficients f_{ij}'s can be computed modulo M^μ without loss of generality.

This set of equations, along with the degree constraints on the coefficients Q_j given by the second condition in Problem 14, precisely gives us an instance of Problem 10, which can also be tackled by choosing the shift $\mathbf{s}$ mentioned above and considering Problem 9.

10.2.2 The general case $d \geqslant 1$

For $F = Y - L$, the reduction presented above can be summarized as follows: first, decompose Q in the basis $\{1, F, \ldots, F^\mu, YF, \ldots, Y^{\lambda-\mu}F^\mu\}$ of $\mathbb{K}[X][Y]_{\leqslant\lambda}$, and second, express the fact that the coefficients in this decomposition must vanish modulo some powers of M, so that Q is actually in the module generated by $\{M^\mu, M^{\mu-1}F, \ldots, MF^{\mu-1}, F^\mu\}$. In this section, we extend this reduction to the general case $d \geqslant 1$.

Eq. (10.1) suggests that the change of basis is now from the basis $\{1, Y, \ldots, Y^\lambda\}$ of $\mathbb{K}[X][Y]_{\leqslant\lambda}$ to its basis

$$\mathcal{E}' \;=\; \{Y^k F^i,\ k < d, i < \mu\} \;\cup\; \{Y^k F^\mu, k \leqslant \lambda - \mu\}.$$

In this more general context, we will not rely on an explicit formula for the coefficients $[\widehat{Q}_i]_i$ of Q in $\mathcal{E}'$. Rather, we will express them as linear combinations of $[Q_j]_j$, and show that the coefficients in these combinations can be computed efficiently. Let us make this more precise by properly defining the problem we are faced with (Problem 17).

Problem 17 – *Change of basis*

Input:

- $M \in \mathbb{K}[X]$ of degree ν,
- $F \in \mathbb{K}[X][Y]$ monic of degree d with coefficients in $\mathbb{K}[X]$ of degree $< \nu$,
- λ, μ positive integers,

Output:

- polynomials $\{f_{ij}^{(k)} \in \mathbb{K}[X]/\langle M^\mu\rangle,\ i < \mu, j \leqslant \lambda, k < d\}$ such that for each $j \leqslant \lambda$, the vector $[f_{ij}^{(k)}]_{i,k}$ gives the first μd coefficients in the decomposition of Y^j in the basis $\mathcal{E}'$:

$$Y^j = \sum_{i<\mu}\sum_{k<d} f_{ij}^{(k)} Y^k F^i + \hat{f}_j F^\mu \ \text{ for some } \hat{f}_j \in \mathbb{K}[X]/\langle M^\mu\rangle[Y]\,. \quad (10.3)$$

Before working on solving Problem 17 efficiently, we explain our interest in this problem: its solution helps us to rewrite the fact that Q belongs to the ideal $\mathcal{I}$ into a set of divisibility properties involving the coefficients $\widehat{Q}_i$ for $i < \mu$. These divisibility properties give us the approximation equations we are looking for.

Lemma 10.2. *Let $Q = Q_0 + Q_1 Y + \cdots + Q_\lambda Y^\lambda$ in $\mathbb{K}[X]/\langle M^\mu\rangle[Y]$. Then, the polynomials $\widehat{Q}_0, \widehat{Q}_1, \ldots, \widehat{Q}_{\mu-1}$ defined for all $i < \mu$ by $\widehat{Q}_i = \sum_{k<d}\sum_{j\leqslant\lambda} f_{ij}^{(k)} Q_j Y^k$ satisfy*

$$Q = \widehat{Q}_0 + \widehat{Q}_1 F + \cdots + \widehat{Q}_{\mu-1}F^{\mu-1} + \widehat{Q}F^\mu$$

in $\mathbb{K}[X]/\langle M^\mu\rangle[Y]$, for some $\widehat{Q} \in \mathbb{K}[X]/\langle M^\mu\rangle[Y]$.

Besides, Q is in the ideal $\mathcal{I}$ generated by $\{M^\mu, M^{\mu-1}F, \ldots, MF^{\mu-1}, F^\mu\}$ if and only if for every $i < \mu$ and $k < d$, $M^{\mu-i}$ divides $\widehat{Q}_i^{(k)}$. Equivalently, writing $\widehat{Q}_i^{(k)} = \sum_{j\leqslant\lambda} f_{ij}^{(k)} Q_j$ for all $i < \mu$ and $k < d$, then Q satisfies the system of linear modular equations

$$\sum_{j\leqslant\lambda} f_{ij}^{(k)} Q_j = 0 \bmod M^{\mu-i} \quad \textit{for all } i < \mu \textit{ and } k < d.$$

We note that here we are working modulo M^μ. This is sufficient for our purpose, since all approximation equations we are going to focus on are modulo powers of M which do not exceed μ. Besides, this helps us to keep control of the size of all elements of $\mathbb{K}[X]$ that we manipulate. In the rest of this section, we show how to solve Problem 17 efficiently, namely in a cost bound that is quasi-linear in the number of field elements used to represent the output $\{f_{ij}^{(k)},\ i < \mu, j \leqslant \lambda, k < d\}$.

Proposition 10.3 (Change of basis)**.** *Problem 17 can be solved using $\mathcal{O}\tilde{}(\lambda\mu^2\nu d)$ operations in $\mathbb{K}$.*

Remark 10.4. Our goal is to apply this change of basis to solve Problem 14. In this context, the coefficients Q_j's of Q are unknowns: this is why we focus on computing the coefficients $f_{ij}^{(k)}$'s of the linear combinations $\widehat{Q}_i$. In the case where the Q_j's are known, one may directly compute the $\widehat{Q}_i$'s as well as $\widehat{Q}$ using $\mathcal{O}\tilde{}(\lambda\mu\nu)$ operations in $\mathbb{K}$.

Furthermore, here we are not interested in computing the polynomial $\widehat{Q}$ in Lemma 10.2, although Algorithm 22 could be adapted to include this computation. One reason is that the mere representation of all coefficients $f_{ij}^{(k)}$, if also computing $\widehat{Q}$, uses $\mathcal{O}\tilde{}(\lambda^2\mu\nu)$ field elements, which is beyond our target cost bound $\mathcal{O}\tilde{}(\lambda^{\omega-1}\mu^2\nu d)$ for solving Problem 14. ☕

We will use the definition of the sought coefficients $f_{ij}^{(k)}$ to compute them incrementally for $j \in \{0, \ldots, \lambda\}$, using the relations between the decomposition of Y^j and $Y^{j+1} = Y \cdot Y^j$. Thus, we first focus on how to use the knowledge of the decomposition of some $P \in \mathbb{K}[X]/\langle M^\mu\rangle[Y]_{<\mu d}$ in $\mathcal{E}'$ to compute the decomposition of YP in $\mathcal{E}'$.

Lemma 10.5. *Write $F = F_0 + \cdots + F_{d-1}Y^{d-1} + Y^d$, with the coefficients $F_0, F_1, \ldots, F_{d-1}$ of degree less than ν. Let $P \in \mathbb{K}[X]/\langle M^\mu\rangle[Y]$ of degree less than μd with coefficients $P_i^{(k)} \in \mathbb{K}[X]/\langle M^\mu\rangle$ in $\mathcal{E}'$, that is,*

$$P \;=\; \sum_{i<\mu}\sum_{k<d} P_i^{(k)} Y^k F^i.$$

Then, Algorithm 21 computes $\{g_i^{(k)},\ i < \mu, k < d\}$ in $\mathbb{K}[X]/\langle M^\mu\rangle$ such that

$$YP \;=\; \sum_{i<\mu}\sum_{k<d} g_i^{(k)} Y^k F^i \;+\; \hat{g}F^\mu \quad \textit{for some } \hat{g} \in \mathbb{K}[X]/\langle M^\mu\rangle \tag{10.4}$$

using $\mathcal{O}\tilde{}(\mu^2\nu d)$ operations in $\mathbb{K}$.

Proof. Using $Y^d = F - F_0 - F_1Y - \cdots - F_{d-1}Y^{d-1}$, we have

$$\begin{aligned}
YP \;&=\; \sum_{i<\mu}\left(P_i^{(d-1)}Y^d + \sum_{0<k<d} P_i^{(k-1)}Y^k\right)F^i \\
&=\; \sum_{i<\mu}\left(P_i^{(d-1)}(F - F_0 - \cdots - F_{d-1}Y^{d-1}) + \sum_{0<k<d} P_i^{(k-1)}Y^k\right)F^i \\
&=\; \sum_{i<\mu}\left(-F_0P_i^{(d-1)} + \sum_{0<k<d}(P_i^{(k-1)} - F_kP_i^{(d-1)})Y^k\right)F^i \\
&\qquad\qquad + \sum_{0<i<\mu} P_{i-1}^{(d-1)}F^i \;+\; P_{\mu-1}^{(d-1)}F^\mu,
\end{aligned}$$

which gives the correctness of Algorithm 21. Now, for each $i < \mu$ and $k < d$ the computation of $g_i^{(k)}$ involves one subtraction and one multiplication in $\mathbb{K}[X]/\langle M^\mu \rangle$. Altogether Step 1 to Step 3 do at most μd subtractions and μd multiplications in $\mathbb{K}[X]/\langle M^\mu \rangle$, which can be done using $\tilde{\mathcal{O}}(\mu^2 \nu d)$ operations in $\mathbb{K}$. ☙

Algorithm 21 – ChangeBasisOneStep
(Change of basis in Coppersmith: from P to YP)

Input:

- $M \in \mathbb{K}[X]$ of degree ν,
- $F = F_0 + \cdots + F_{d-1}Y^{d-1} + Y^d$ in $\mathbb{K}[X][Y]$ with $\deg(F_k) < \nu$ for all k,
- positive integer μ,
- $P = \sum_{i<\mu} \sum_{k<d} P_i^{(k)} Y^k F^i$ in $\mathbb{K}[X]/\langle M^\mu \rangle[Y]$.

Output: $\{g_i^{(k)},\ i < \mu, k < d\}$ in $\mathbb{K}[X]/\langle M^\mu \rangle$ such that Eq. (10.4).

1. $g_0^{(0)} \leftarrow -F_0 P_0^{(d-1)}$; $g_i^{(0)} \leftarrow P_{i-1}^{(d-1)} - F_0 P_i^{(d-1)}$ for $0 < i < \mu$
2. $g_i^{(k)} \leftarrow P_i^{(k-1)} - F_k P_i^{(d-1)}$ for $0 < k < d, 0 < i < \mu$
3. Return $\{g_i^{(k)},\ i < \mu, k < d\}$

We now have all the tools to give our main algorithm, which proves Proposition 10.3.

Proof (Proposition 10.3). Algorithm 22 finds the polynomials $\{f_{ij}^{(k)}, j \leqslant \lambda, i < \mu, k < d\}$, starting from the $F_{0i}^{(k)}$'s that are known and using λ calls to Algorithm 21. Its cost is thus bounded by $\tilde{\mathcal{O}}(\lambda \mu^2 \nu d)$, and its correctness follows from that of Algorithm 21. ☙

Algorithm 22 – ChangeBasis
(Change of basis in Coppersmith technique)

Input:

- $M \in \mathbb{K}[X]$ of degree ν,
- $F \in \mathbb{K}[X][Y]$ monic of degree d with coefficients of degree less than ν,
- positive integers μ, λ.

Output: $\{f_{ij}^{(k)}, j \leqslant \lambda, i < \mu, k < d\}$ in $\mathbb{K}[X]/\langle M^\mu \rangle$ as in Eq. (10.3) for $j \leqslant \lambda$.

1. Set $F_{00}^{(0)} \leftarrow 1$ and $F_{0i}^{(k)} \leftarrow 0$ for $i > 0$ or $k > 0$
2. For j from 1 to λ do
   a. Define $P = \sum_{i<\mu} \sum_{k<d} F_{j-1,i}^{(k)} Y^k$
   b. $\{f_{ij}^{(k)},\ i < \mu, k < d\} \leftarrow$ Algorithm 21 on input M, F, μ, P
3. Return $\{f_{ij}^{(k)},\ j \leqslant \lambda, i < \mu, k < d\}$

Part IV

Interpolant bases and multivariate interpolation

Contents

Chapter 11 Multivariate interpolation and list-decoding **229**

11.1 Reducing Problem 11 to Problem 10 229

11.2 Impact on decoding algorithms for Reed-Solomon codes 234

11.2.1 Interpolation step of the Guruswami-Sudan algorithm 234

11.2.2 Re-encoding technique 235

11.2.3 Interpolation step in the Wu algorithm 237

11.2.4 Slowdown due to repeating points in the soft-decoding 239

11.3 The approach based on row reduction 240

11.4 On assumption $\mathcal{H}_{\text{int},1}$ 242

11.5 On assumption $\mathcal{H}_{\text{int},3}$ 243

Chapter 12 Some tools for computing with polynomial matrices **245**

12.1 More time functions for polynomial matrices 245

12.2 Multiplying matrices with unbalanced row degrees [ZLS12] 248

12.3 Detailed cost bound for the kernel basis algorithm of [ZLS12] 249

Chapter 13 Computing shifted Popov interpolant bases **255**

13.1 Divide-and-conquer approach for a triangular multiplication matrix . . 255

13.2 Fast interpolant bases in reduced form for almost uniform shifts 257

13.3 Fast interpolant bases in Popov form for arbitrary shifts 261

Chapter 14 Details of new ingredients for interpolant bases **265**

14.1 Fast shifted reduction of a reduced matrix 265

14.2 Computing residuals for interpolant bases 267

14.2.1 Residuals and Chinese remaindering 268

14.2.2 Main algorithm 270
14.2.3 Computing the residual by shifting **P** 272
14.2.4 Computing the residual by Chinese remaindering 274
14.3 Computing interpolant bases with known minimal degree 277

11

Multivariate interpolation and list-decoding

In this chapter, we tackle the first of our two problems of multivariate interpolation (Problem 11). It essentially asks to compute a single multivariate interpolant with prescribed multiplicities and degree constraints, under the assumption that the X-coordinates of the points are pairwise distinct.

We start by reducing this problem to a system of linear modular equation; the assumption on the points ensures that the number of polynomial equations is at most the number of unknowns. In this situation, we can rely on our results exposed above (Theorems 2.22 and 2.25) to find the sought interpolant efficiently. Then, we give details of consequences on list-decoding algorithms for Reed-Solomon codes: we discuss the original Guruswami-Sudan algorithm, the Wu algorithm, and the use of the re-encoding technique in combination with our algorithms.

After this, we observe that the assumption on the points is not satisfied in the context of the soft-decoding of Reed-Solomon codes. This implies that the system of modular equations obtained after reduction may involve significantly more equations than unknowns, a case that our results do not cover with the target efficiency.

This is one of our motivations for studying interpolant bases, in Chapters 12 to 14. We recall that interpolant bases correspond to systems of modular equations where we know the roots and multiplicities of the moduli: in this case, our results do not involve any requirement on these roots (Theorems 2.19 and 2.20).

11.1 Reducing Problem 11 to Problem 10

In this section, we show how instances of Problem 11 can be reduced to instances of Problem 10; Algorithm 23 gives an overview of this reduction. The main technical ingredient, stated in Lemma 11.2 below, generalizes to any $r \geqslant 1$ and (possibly) distinct multiplicities the result given for $r = 1$ by Zeh, Gentner, and Augot in [ZGA11, Proposition 3]. To prove it, we use the same steps as in [ZGA11]; we rely on the notion of Hasse derivatives, which allows us to write Taylor expansions in positive characteristic (see Hasse [Has36] or Roth [Rot07, pp. 87, 276]).

For better readability, in what follows we use italicized boldface letters to denote r-tuples of objects that are related to Y variables: $\boldsymbol{Y}^{\boldsymbol{i}} = Y_1^{i_1} \cdots Y_r^{i_r}$, $\boldsymbol{w} = (w_1, \ldots, w_r)$, etc. In the special case of r-tuples of integers, we also write $|\boldsymbol{w}| = w_1 + \cdots + w_r$. Comparison and addition of multi-indices in $\mathbb{Z}_{\geqslant 0}^r$ are defined componentwise. For example, writing $\boldsymbol{i} \leqslant \boldsymbol{j}$ is equivalent to having $i_1 \leqslant j_1, \ldots, i_r \leqslant j_r$ simultaneously, $\boldsymbol{i} - \boldsymbol{j}$ denotes $(i_1 - j_1, \ldots, i_r - j_r)$, and for $\boldsymbol{a} = (a_1, \ldots, a_r) \in \mathbb{K}[X]^r$ we let $(\boldsymbol{Y} - \boldsymbol{a})^{\boldsymbol{i}} = (Y_1 - a_1)^{i_1} \cdots (Y_r - a_r)^{i_r}$. Besides, for products of binomial coefficients, we use the notation

$$\binom{\boldsymbol{j}}{\boldsymbol{i}} = \binom{j_1}{i_1} \cdots \binom{j_r}{i_r};$$

note that this integer is zero when $\boldsymbol{i} \not\leqslant \boldsymbol{j}$.

If $\mathcal{R}$ is any commutative ring with unity and $\mathcal{R}[\boldsymbol{Y}]$ denotes the ring of polynomials in $Y_1, \ldots, Y_r$ over $\mathcal{R}$, then for a polynomial $Q(\boldsymbol{Y}) = \sum_{\boldsymbol{j}} Q_{\boldsymbol{j}} \boldsymbol{Y}^{\boldsymbol{j}} \in \mathcal{R}[\boldsymbol{Y}]$ and a multi-index $\boldsymbol{i}$ in $\mathbb{Z}_{\geqslant 0}^r$, the *order-$\boldsymbol{i}$ Hasse derivative of* Q is the polynomial $Q^{[\boldsymbol{i}]}$ in $\mathcal{R}[\boldsymbol{Y}]$ defined by

$$Q^{[\boldsymbol{i}]} = \sum_{\boldsymbol{j} \geqslant \boldsymbol{i}} \binom{\boldsymbol{j}}{\boldsymbol{i}} Q_{\boldsymbol{j}} \boldsymbol{Y}^{\boldsymbol{j} - \boldsymbol{i}}.$$

The Hasse derivative satisfies the following property (Taylor expansion): for all $\boldsymbol{a}$ in $\mathcal{R}^r$,

$$Q(\boldsymbol{Y}) = \sum_{\boldsymbol{i}} Q^{[\boldsymbol{i}]}(\boldsymbol{a})(\boldsymbol{Y} - \boldsymbol{a})^{\boldsymbol{i}}.$$

The next lemma shows how Hasse derivatives help rephrase the vanishing condition *(iii)* of Problem 11 for one of the points $\{(x_k, \boldsymbol{y}_k)\}_{1 \leqslant k \leqslant \nu}$.

Lemma 11.1. *Let $(x, y_1, \ldots, y_r)$ be a point in $\mathbb{K}^{r+1}$ and $\boldsymbol{L} = (L_1, \ldots, L_r) \in \mathbb{K}[X]^r$ be such that $L_j(x) = y_j$ for $1 \leqslant j \leqslant r$. Then, for any polynomial Q in $\mathbb{K}[X, \boldsymbol{Y}]$, $Q(x, \boldsymbol{y}) = 0$ with multiplicity at least μ if and only if for all $\boldsymbol{i}$ in $\mathbb{Z}_{\geqslant 0}^r$ such that $|\boldsymbol{i}| < \mu$,*

$$Q^{[\boldsymbol{i}]}(X, \boldsymbol{L}) = 0 \bmod (X - x)^{\mu - |\boldsymbol{i}|}.$$

Proof. Up to a shift, one can assume that the point is $(x, y_1, \ldots, y_r) = (0, \boldsymbol{0})$; in other words, it suffices to show that for $\boldsymbol{L}(0) = \boldsymbol{0} \in \mathbb{K}^r$, we have $Q(0, \boldsymbol{0}) = 0$ with multiplicity at least μ if and only if, for all $\boldsymbol{i}$ in $\mathbb{Z}_{\geqslant 0}^r$ such that $|\boldsymbol{i}| < \mu$, $X^{\mu - |\boldsymbol{i}|}$ divides $Q^{[\boldsymbol{i}]}(X, \boldsymbol{L})$.

Assume first that $(0, \boldsymbol{0}) \in \mathbb{K}^{r+1}$ is a root of Q of multiplicity at least μ. Then, $Q(X, \boldsymbol{Y}) = \sum_{\boldsymbol{j}} Q_{\boldsymbol{j}} \boldsymbol{Y}^{\boldsymbol{j}}$ has only monomials of total degree at least μ, so that for $\boldsymbol{j} \geqslant \boldsymbol{i}$, each nonzero $Q_{\boldsymbol{j}} \boldsymbol{Y}^{\boldsymbol{j} - \boldsymbol{i}}$ has only monomials of total degree at least $\mu - |\boldsymbol{i}|$. Now, $\boldsymbol{L}(0) = \boldsymbol{0} \in \mathbb{K}^r$ implies that X divides each component of $\boldsymbol{L}$. Consequently, $X^{\mu - |\boldsymbol{i}|}$ divides $Q_{\boldsymbol{j}} \boldsymbol{L}^{\boldsymbol{j} - \boldsymbol{i}}$ for each $\boldsymbol{j} \geqslant \boldsymbol{i}$, and thus $Q^{[\boldsymbol{i}]}(X, \boldsymbol{L})$ as well.

Conversely, let us assume that for all $\boldsymbol{i}$ in $\mathbb{Z}_{\geqslant 0}^r$ such that $|\boldsymbol{i}| < \mu$, $X^{\mu - |\boldsymbol{i}|}$ divides $Q^{[\boldsymbol{i}]}(X, \boldsymbol{L})$, and show that Q has no monomial of total degree less than μ. Writing the Taylor expansion of Q with $\mathbb{K}[\boldsymbol{X}] = \mathbb{K}[X]$ and $\boldsymbol{a} = \boldsymbol{L}$, we obtain

$$Q(X, \boldsymbol{Y}) = \sum_{\boldsymbol{i}} Q^{[\boldsymbol{i}]}(X, \boldsymbol{L})(\boldsymbol{Y} - \boldsymbol{L})^{\boldsymbol{i}}.$$

Each component of $\boldsymbol{L}$ being a multiple of X, we deduce that for the multi-indices $\boldsymbol{i}$ such that $|\boldsymbol{i}| \geqslant \mu$ every nonzero monomial in $Q^{[\boldsymbol{i}]}(X, \boldsymbol{L})(\boldsymbol{Y} - \boldsymbol{L})^{\boldsymbol{i}}$ has total degree at least μ. Using our assumption, the same conclusion follows for the multi-indices such that $|\boldsymbol{i}| < \mu$. ☙

Thus, for each of the points $\{(x_k, \boldsymbol{y}_k)\}_{1 \leqslant k \leqslant \nu}$ in Problem 11, such a rewriting of the vanishing condition *(iii)* for this point holds. Now the fact that the x_i are distinct intervenes: the polynomials $(X - x_u)^\alpha$ and $(X - x_v)^\beta$ are coprime for $u \neq v$, so that simultaneous divisibility by both those polynomials is equivalent to divisibility by their product $(X - x_u)^\alpha (X - x_v)^\beta$. Using the r-tuple $\boldsymbol{L} = (L_1, \ldots, L_r) \in \mathbb{K}[X]^r$ of Lagrange interpolation polynomials, defined by the conditions

$$\deg(L_j) < \nu \quad \text{and} \quad L_j(x_i) = y_{i,j} \quad \text{for} \quad 1 \leqslant i \leqslant \nu \quad \text{and} \quad 1 \leqslant j \leqslant r, \tag{11.1}$$

we can then combine Lemma 11.1 for all points so as to rewrite the vanishing condition of Problem 11 as a set of modular equations in $\mathbb{K}[X]$ as in Lemma 11.2 below. In what follows, we use the notation from Problem 11 and Corollary 3.1.

Lemma 11.2. *For any polynomial Q in $\mathbb{K}[X, \boldsymbol{Y}]$, Q satisfies the condition (iii) of Problem 11 if and only if for all $\boldsymbol{i}$ in $\mathbb{Z}^r_{\geqslant 0}$ such that $|\boldsymbol{i}| < \mu$,*

$$Q^{[\boldsymbol{i}]}(X, \boldsymbol{L}) = 0 \mod \prod_{\substack{1 \leqslant k \leqslant \nu: \\ \mu_k > |\boldsymbol{i}|}} (X - x_k)^{\mu_k - |\boldsymbol{i}|}.$$

Proof. This result follows from Lemma 11.1 since the x_k are pairwise distinct. ☙

Note that when all multiplicities are equal, that is, $\mu = \mu_1 = \cdots = \mu_\nu$, for every $|\boldsymbol{i}|$ the modulus takes the simpler form $M^{\mu - |\boldsymbol{i}|}$, where $M = \prod_{1 \leqslant k \leqslant \nu}(X - x_k)$.

Writing $\boldsymbol{j} \cdot \boldsymbol{w} = j_1 w_1 + \cdots + j_s w_r$, recall from the statement of Corollary 3.1 that $\mathcal{S}$ is the set of all $\boldsymbol{j}$ in $\mathbb{Z}^s_{\geqslant 0}$ such that $|\boldsymbol{j}| \leqslant \lambda$ and $\boldsymbol{j} \cdot \boldsymbol{w} < b$. Then, defining the positive integers

$$N_{\boldsymbol{j}} = b - \boldsymbol{j} \cdot \boldsymbol{w}$$

for all $\boldsymbol{j}$ in $\mathcal{S}$, we immediately obtain the following reformulation of the list-size and weighted-degree conditions of our interpolation problem:

Lemma 11.3. *For any polynomial Q in $\mathbb{K}[X, \boldsymbol{Y}]$, Q satisfies the conditions (i) and (ii) of Problem 11 if and only if it has the form*

$$Q(X, \boldsymbol{Y}) = \sum_{\boldsymbol{j} \in \mathcal{S}} Q_{\boldsymbol{j}}(X) \boldsymbol{Y}^{\boldsymbol{j}} \quad \textit{with} \quad \deg(Q_{\boldsymbol{j}}) < N_{\boldsymbol{j}}.$$

For $\boldsymbol{i} \in \mathbb{Z}^r_{\geqslant 0}$ with $|\boldsymbol{i}| < \mu$ and $\boldsymbol{j} \in \mathcal{S}$, let us now define the polynomials $\mathfrak{m}_{\boldsymbol{i}}, F_{\boldsymbol{i},\boldsymbol{j}} \in \mathbb{K}[X]$ as

$$\mathfrak{m}_{\boldsymbol{i}} = \prod_{\substack{1 \leqslant k \leqslant \nu: \\ \mu_k > |\boldsymbol{i}|}} (X - x_k)^{\mu_k - |\boldsymbol{i}|} \tag{11.2a}$$

and

$$F_{\boldsymbol{i},\boldsymbol{j}} = \binom{\boldsymbol{j}}{\boldsymbol{i}} \boldsymbol{L}^{\boldsymbol{j}-\boldsymbol{i}} \bmod \mathfrak{m}_{\boldsymbol{i}}. \tag{11.2b}$$

It then follows from Lemmas 11.2 and 11.3 that $Q \in \mathbb{K}[X, \boldsymbol{Y}]$ satisfies the conditions *(i)*, *(ii)*, and *(iii)* of Problem 11 if and only if $Q = \sum_{\boldsymbol{j} \in \mathcal{S}} Q_{\boldsymbol{j}} \boldsymbol{Y}^{\boldsymbol{j}}$ for some polynomials $Q_{\boldsymbol{j}} \in \mathbb{K}[X]$ such that

- $\deg(Q_{\boldsymbol{j}}) < N_{\boldsymbol{j}}$ for all $\boldsymbol{j}$ in $\mathcal{S}$,
- $\sum_{\boldsymbol{j} \in \mathcal{S}} F_{\boldsymbol{i},\boldsymbol{j}} Q_{\boldsymbol{j}} = 0 \bmod \mathfrak{m}_{\boldsymbol{i}}$ for all $|\boldsymbol{i}| < \mu$.

Let now $D_{\boldsymbol{i}}$ be the positive integers given by

$$D_{\boldsymbol{i}} = \sum_{1 \leqslant k \leqslant \nu \colon \mu_k > |\boldsymbol{i}|} (\mu_k - |\boldsymbol{i}|),$$

for all $|\boldsymbol{i}| < \mu$. Since the $\mathfrak{m}_{\boldsymbol{i}}$ are monic polynomials of degree $D_{\boldsymbol{i}}$ and since $\deg F_{\boldsymbol{i},\boldsymbol{j}} < D_{\boldsymbol{i}}$, the latter conditions express the problem of finding such a Q as an instance of Problem 10. In order to make the reduction completely explicit, define further

$$D \;=\; \sum_{|\boldsymbol{i}| < \mu} D_{\boldsymbol{i}}\ ,$$

$$n = \binom{s + \mu - 1}{s}, \qquad m = \mathrm{Card}(\mathcal{S}), \qquad \rho = \max(n, m);$$

then choose arbitrary orders on the sets of indices $\{\boldsymbol{i} \in \mathbb{Z}^s_{\geqslant 0} \mid |\boldsymbol{i}| < \mu\}$ and $\mathcal{S}$, that is, bijections

$$\phi : \{0, \ldots, n-1\} \to \{\boldsymbol{i} \in \mathbb{Z}^s_{\geqslant 0} \mid |\boldsymbol{i}| < \mu\} \tag{11.3a}$$

and

$$\psi : \{0, \ldots, m-1\} \to \mathcal{S}; \tag{11.3b}$$

finally, for i in $\{0, \ldots, n-1\}$ and j in $\{0, \ldots, m-1\}$, associate $D'_i = D_{\phi(i)}$, $N'_j = N_{\psi(j)}$, $\mathfrak{m}'_i = \mathfrak{m}_{\phi(i)}$ and $F'_{i,j} = F_{\phi(i),\psi(j)}$. At this stage, we have showed that the set of solutions to Problem 11 with input parameters $r, \lambda, \nu, \mu_1, \ldots, \mu_\nu, b, w_1, \ldots, w_r$ and points $\{(x_k, y_{k,1}, \ldots, y_{k,r})\}_{1 \leqslant k \leqslant \nu}$ is exactly the set of solutions to Problem 10 with as input the moduli $(\mathfrak{m}'_i)_{0 \leqslant i < n}$, the matrix $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ whose columns are $[F'_{i,0}, \ldots, F'_{i,m-1}]^{\mathsf{T}}$ for $0 \leqslant i < n$, and the degree constraints $N_{\psi(0)}, \ldots, N_{\psi(m-1)}$. This proves the correctness of Algorithm 23.

Proposition 11.4. *Algorithm 23 is correct and uses*

$$\mathcal{O}((m+n)\,\mathsf{M}(D)\log(D))$$

operations in $\mathbb{K}$.

Algorithm 23 – MultiIntViaSolVec
(Finding multivariate interpolants as solution vectors)

Input:

- $r, \lambda, \nu, \mu_1, \ldots, \mu_\nu$ in $\mathbb{Z}_{>0}$,
- $b, w_1, \ldots, w_r$ in $\mathbb{Z}$,
- points $\{(x_k, y_{k,1}, \ldots, y_{k,r})\}_{1 \leqslant k \leqslant \nu}$ in $\mathbb{K}^{r+1}$ with the x_k pairwise distinct.

Output: an instance $(\mathfrak{M}, \mathbf{F}, \mathbf{N})$ of Problem 10, such that the set of solutions to this instance is exactly the set of solutions to Problem 11 with parameters the input of this algorithm.

1. Compute $\mathcal{S} = \{\boldsymbol{j} \in \mathbb{Z}^r_{\geqslant 0} \mid |\boldsymbol{j}| \leqslant \lambda \text{ and } b - \boldsymbol{j} \cdot \boldsymbol{w} > 0\}$, $n = \binom{r+\mu-1}{r}$, $m = \mathrm{Card}(\mathcal{S})$, and bijections ϕ and ψ as in Eq. (11.3)
2. Compute $D_{\boldsymbol{i}} = \sum_{1 \leqslant k \leqslant \nu:\, \mu_k > |\boldsymbol{i}|} (\mu_k - |\boldsymbol{i}|)$ and $N_{\boldsymbol{j}} = b - \boldsymbol{j} \cdot \boldsymbol{w}$ for $\boldsymbol{j} \in \mathcal{S}$
3. Compute $\mathfrak{m}_{\boldsymbol{i}}$ and $F_{\boldsymbol{i},\boldsymbol{j}}$ for $|\boldsymbol{i}| < \mu$, $\boldsymbol{j} \in \mathcal{S}$ as in Eq. (11.2)
4. Return the moduli $(\mathfrak{m}_{\phi(i)})_{0 \leqslant i < n}$, the matrix $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ whose columns are $[F_{\phi(i),\psi(0)}, \ldots, F_{\phi(i),\psi(m-1)}]^\mathsf{T}$ for $0 \leqslant i < n$, and the degree constraints $N_{\psi(0)}, \ldots, N_{\psi(m-1)}$

Proof. The only thing left to do is the complexity analysis; more precisely, giving an upper bound on the number of operations in $\mathbb{K}$ performed in Step **3**.

First, we need to compute $\mathfrak{m}_{\boldsymbol{i}}$ as in (11.2a) for every $\boldsymbol{i} \in \mathbb{Z}^r_{\geqslant 0}$ such that $|\boldsymbol{i}| < \mu$. This involves only μ different polynomials $\mathfrak{m}_{\boldsymbol{i}_0}, \ldots, \mathfrak{m}_{\boldsymbol{i}_{\mu-1}}$ where we have chosen any indices $\boldsymbol{i}_j$ such that $|\boldsymbol{i}_j| = j$. We note that, defining $\mathfrak{n}_j = \prod_{1 \leqslant k \leqslant \nu:\, \mu_k > j} (X - x_k)$ for $j < \mu$, we have $\mathfrak{m}_{\boldsymbol{i}_{\mu-1}} = \mathfrak{n}_{\mu-1}$ and $\mathfrak{m}_{\boldsymbol{i}_j} = \mathfrak{m}_{\boldsymbol{i}_{j+1}} \mathfrak{n}_j$ for $j < \mu - 1$. The polynomials $\mathfrak{n}_0, \ldots, \mathfrak{n}_{\mu-1}$ have degree at most ν and can be computed using $\mathcal{O}(\mu \mathsf{M}(\nu) \log(\nu))$ operations in $\mathbb{K}$; for $\rho = \max(m, n)$, this is in $\mathcal{O}(\rho \mathsf{M}(D) \log(D))$ since $\rho \geqslant \binom{r+\mu-1}{r} \geqslant \mu$ and $D = \sum_{1 \leqslant k \leqslant \nu} \binom{r+\mu_k}{r+1} \geqslant \nu$. Then $\mathfrak{m}_{\boldsymbol{i}_0}, \ldots, \mathfrak{m}_{\boldsymbol{i}_{\mu-1}}$ can be computed iteratively using $\mathcal{O}(\sum_{j<\mu} \mathsf{M}(\deg(\mathfrak{m}_{\boldsymbol{i}_j})))$ operations in $\mathbb{K}$; using the super-linearity of $\mathsf{M}(\cdot)$ in Eq. (6.1), this is in $\mathcal{O}(\mathsf{M}(D))$ since $\deg(\mathfrak{m}_{\boldsymbol{i}_j}) = D_{\boldsymbol{i}_j}$ and $\sum_{j<\mu} D_{\boldsymbol{i}_j} \leqslant D$.

Then, we have to compute (some of) the interpolation polynomials $L_1, \ldots, L_r$. Due to Lemma 11.2, the only values of $i \in \{1, \ldots, r\}$ for which L_i is needed are those such that the indeterminate Y_i may actually appear in $Q(X, \boldsymbol{Y}) = \sum_{\boldsymbol{j} \in \mathcal{S}} Q_{\boldsymbol{j}}(X) \boldsymbol{Y}^{\boldsymbol{j}}$. The latter will not occur unless the ith unit r-tuple $(0, \ldots, 0, 1, 0, \ldots, 0)$ belongs to $\mathcal{S}$. Hence, at most $\mathrm{Card}(\mathcal{S})$ polynomials L_i must be computed, each at a cost of $\mathcal{O}(\mathsf{M}(\nu) \log(\nu))$ operations in $\mathbb{K}$. Overall, the cost of the interpolation step is thus in $\mathcal{O}(\mathrm{Card}(\mathcal{S}) \mathsf{M}(\nu) \log(\nu)) \subseteq \mathcal{O}(\rho\, \mathsf{M}(D) \log(D))$.

Finally, we compute $F_{\boldsymbol{i},\boldsymbol{j}}$ as in (11.2b) for every $\boldsymbol{i}, \boldsymbol{j}$. This is done by fixing $\boldsymbol{i}$ and computing all products $F_{\boldsymbol{i},\boldsymbol{j}}$ incrementally, starting from $L_1, \ldots, L_r$. Since we compute modulo $\mathfrak{m}_{\boldsymbol{i}}$, each product takes $\mathcal{O}(\mathsf{M}(D_{\boldsymbol{i}}))$ operations in $\mathbb{K}$. Summing over all $\boldsymbol{j}$ leads to a cost of $\mathcal{O}(\mathrm{Card}(\mathcal{S}) \mathsf{M}(D_{\boldsymbol{i}}))$ per index $\boldsymbol{i}$. Summing over all $\boldsymbol{i}$ and using the super-linearity of $\mathsf{M}(\cdot)$ leads to a total cost of $\mathcal{O}(\mathrm{Card}(\mathcal{S}) \mathsf{M}(D))$, which is $\mathcal{O}(\rho \mathsf{M}(D))$. ∎

The reduction above is deterministic and its cost is negligible compared to the cost in $\mathcal{O}((m+n)^{\omega-1}\mathsf{M}(D)\log(D))$ that follows from Theorem 2.25 with $D = \sum_{0\leqslant i<n} D_i = D$. Noting that $D = \sum_{|\boldsymbol{i}|<\mu} D_{\boldsymbol{i}} = \sum_{1\leqslant k\leqslant \nu} \binom{r+\mu_k}{r+1}$, we conclude that Theorem 2.25 implies Corollary 3.1.

11.2 Impact on decoding algorithms for Reed-Solomon codes

In this section, we discuss Corollary 3.1 in specific contexts related to the decoding of Reed-Solomon codes; in particular, we always have $r = 1$. First, we give our complexity result in the case of list-decoding via the Guruswami-Sudan algorithm [GS99]; then we show how the re-encoding technique [KV03b, KMV11] can be used in our setting; then, we discuss the interpolation step of the Wu algorithm [Wu08]; and finally we present the application of our results to the interpolation step of the soft-decoding [KV03a]. In these contexts of applications, we will use some of the assumptions on the parameters $\mathcal{H}_{\text{int},1}$, $\mathcal{H}_{\text{int},2}$, $\mathcal{H}_{\text{int},3}$, $\mathcal{H}_{\text{int},4}$ given in Section 3.1.1. Note that in the context of soft-decoding, the x_i in the input of Problem 11 are not necessarily pairwise distinct: we will explain how to adapt our algorithms to this case. Besides, still in this context, the number of points ν is no longer equal to the length of the code and may actually be much larger, unlike in hard-decision (list-)decoding.

11.2.1 Interpolation step of the Guruswami-Sudan algorithm

We study here the specific context of the interpolation step of the Guruswami-Sudan list-decoding algorithm for Reed-Solomon codes. This interpolation step is precisely Problem 11 where we have $r = 1$ and we make assumptions $\mathcal{H}_{\text{int},1}$, $\mathcal{H}_{\text{int},2}$, $\mathcal{H}_{\text{int},3}$, $\mathcal{H}_{\text{int},4}$. Under $\mathcal{H}_{\text{int},2}$, the set $\mathcal{S}$ introduced in Corollary 3.1 reduces to $\{j \in \mathbb{Z}_{\geqslant 0} : j \leqslant \lambda\} = \{0, \ldots, \lambda\}$, so that $\mathrm{Card}(\mathcal{S}) = \lambda + 1$. Thus, assumption $\mathcal{H}_{\text{int},1}$ ensures that the parameter ρ in that theorem is $\rho = \lambda + 1$; because of $\mathcal{H}_{\text{int},4}$ all multiplicities are equal so that we further have $D = \binom{\mu+1}{2}\nu = \frac{\mu(\mu+1)}{2}\nu$. From Corollary 3.1, we obtain the following result, which substantiates our claimed cost bound in Section 3.1.1, Table 3.1.

Corollary 11.5. *Taking $r = 1$, if the parameters λ, ν, $\mu := \mu_1 = \cdots = \mu_\nu$, b, and $w := w_1$ satisfy $\mathcal{H}_{\text{int},1}$, $\mathcal{H}_{\text{int},2}$, $\mathcal{H}_{\text{int},3}$, $\mathcal{H}_{\text{int},4}$, then there exists a probabilistic algorithm that computes a solution to Problem 11 using*

$$\mathcal{O}(\lambda^{\omega-1}\mathsf{M}(\mu^2\nu)\log(\mu\nu)) \subseteq \mathcal{O}\tilde{}(\lambda^{\omega-1}\mu^2\nu)$$

operations in $\mathbb{K}$, with probability of success at least $1/2$.

We note that the probability analysis in Corollary 3.1 is simplified in this context. Indeed, to ensure probability of success at least $1/2$, the algorithm chooses $\mathcal{O}(\mu^2\nu)$ elements uniformly at random in a subset of $\mathbb{K}$ of cardinality at least $24\mu^4\nu^2$; if $\mathrm{Card}(\mathbb{K}) < 24\mu^4\nu^2$, one can use the remarks following Corollary 3.1 in Section 3.1.1 about solving the problem over an extension of $\mathbb{K}$ and retrieving a solution over $\mathbb{K}$. Here, the base field $\mathbb{K}$

must be of cardinality at least ν since the x_i are pairwise distinct; then, an extension degree $d = \mathcal{O}(\log_\nu(\mu))$ suffices and the cost bound above becomes $\mathcal{O}(\lambda^{\omega-1}\mathsf{M}(\mu^2\nu)\log(\mu\nu) \cdot \mathsf{M}(d)\log(d))$. Besides, in the list-decoding of Reed-Solomon codes we have $\mu = \mathcal{O}(\nu^2)$, so that $d = \mathcal{O}(1)$ and the cost bound and probability of success in Corollary 11.5 hold for *any* field $\mathbb{K}$ (of cardinality at least ν).

11.2.2 Re-encoding technique

The re-encoding technique, introduced by Welch and Berlekamp [WB86] and later extended by Koetter and Vardy [KV03b, KMV11], leads to a reduction of the cost of the interpolation step in list- and soft-decoding of Reed-Solomon codes. Here, for the sake of clarity, we present this technique only in the context of Reed-Solomon list-decoding via the Guruswami-Sudan algorithm, using the same notation and assumptions as in Section 11.2.1 above: $r = 1$ and we have $\mathcal{H}_{\text{int},1}$, $\mathcal{H}_{\text{int},2}$, $\mathcal{H}_{\text{int},3}$, $\mathcal{H}_{\text{int},4}$. Under some additional assumption on the input points in Problem 11, by means of partially pre-solving the problem one obtains an interpolation problem whose linearization has smaller dimensions. The idea at the core of this technique is summarized in the next lemma [KV03b, Lemma 4].

Lemma 11.6. *Let μ be a positive integer, x be an element in $\mathbb{K}$, and $Q = \sum_j Q_j(X)Y^j$ be a polynomial in $\mathbb{K}[X,Y]$. Then, $Q(x,0) = 0$ with multiplicity at least μ if and only if $(X-x)^{\mu-j}$ divides Q_j for all $j < \mu$.*

Proof. By definition, $Q(x,0) = 0$ with multiplicity at least μ if and only if $Q(X+x,Y)$ has no monomial of total degree less than μ. Since $Q(X+x,Y) = \sum_j Q_j(X+x)Y^j$, this is equivalent to the fact that $X^{\mu-j}$ divides $Q_j(X+x)$ for each $j < \mu$. ☛

This property can be generalized to the case of several roots of the form $(x,0)$. More precisely, the re-encoding technique is based on a shift of the received word by a well-chosen code word, which allows us to ensure the following assumption on the points $\{(x_k,y_k)\}_{1\leqslant k\leqslant\nu}$: for some integer $\nu_0 \geqslant w+1$,

$$y_1 = \cdots = y_{\nu_0} = 0 \quad \text{and} \quad y_{\nu_0+1} \neq 0, \ldots, y_\nu \neq 0. \tag{11.4}$$

We now define the polynomial $M_0 = \prod_{1\leqslant k\leqslant\nu_0}(X - x_k)$ which vanishes at x_k when $y_k = 0$, and Lemma 11.6 can be rewritten as follows: $Q(x_k,0) = 0$ with multiplicity at least μ for $1 \leqslant k \leqslant \nu_0$ if and only if $M_0^{\mu-j}$ divides Q_j for each $j < \mu$. Thus, we know how to solve the vanishing condition for the ν_0 points for which $y_k = 0$: by setting each of the μ polynomials $Q_0, \ldots, Q_{\mu-1}$ as the product of a power of M_0 and an unknown polynomial. Combining this with the polynomial approximation problem corresponding to the points $\{(x_k,y_k)\}_{\nu_0+1\leqslant k\leqslant\nu}$, there remains to solve a smaller approximation problem.

Indeed, under the previously mentioned assumptions $r = 1$ and $\mathcal{H}_{\text{int},1}$, $\mathcal{H}_{\text{int},2}$, $\mathcal{H}_{\text{int},3}$, $\mathcal{H}_{\text{int},4}$, it has been shown in Section 11.1 that the vanishing condition *(iii)* of Problem 11 restricted to points $\{(x_k,y_k)\}_{\nu_0+1\leqslant k\leqslant\nu}$ is equivalent to the simultaneous polynomial approximations

$$\sum_{i\leqslant j\leqslant\lambda} \binom{j}{i} L^{j-i} Q_j = 0 \mod M^{m-i} \qquad \text{for } i < \mu,$$

where $M = \prod_{\nu_0+1 \leqslant k \leqslant \nu}(X - x_k)$ and L is the interpolation polynomial such that $\deg(L) < \nu - \nu_0$ and $L(x_k) = y_k$ for $\nu_0 + 1 \leqslant k \leqslant \nu$. On the other hand, we have seen that the vanishing condition for the points $\{(x_k, y_k)\}_{1 \leqslant k \leqslant \nu_0}$ is equivalent to $Q_j = M_0^{\mu-j} Q_j^\star$ for each $j < \mu$, for some unknown polynomials $Q_0^\star, \ldots, Q_{\mu-1}^\star$. Combining both equivalences, we obtain for $i < \mu$

$$\sum_{i \leqslant j < \mu} F_{i,j}\, Q_j^\star + \sum_{\mu \leqslant j \leqslant \lambda} F_{i,j}\, Q_j = 0 \mod M^{\mu-i} \tag{11.5}$$

with

$$F_{i,j} = \begin{cases} \binom{j}{i} L^{j-i} M_0^{\mu-j} \bmod M^{\mu-i} & \text{for } i \leqslant j < \mu, \\ \binom{j}{i} L^{j-i} \bmod M^{\mu-i} & \text{for } \mu \leqslant j \leqslant \lambda. \end{cases} \tag{11.6}$$

Obviously, the degree constraints on $Q_0, \ldots, Q_{\mu-1}$ directly correspond to degree constraints on $Q_0^\star, \ldots, Q_{\mu-1}^\star$ while those on $Q_\mu, \ldots, Q_\lambda$ are unchanged. The number of equations obtained when linearizing Eq. (11.5) is $D = \sum_{i<\mu} \deg(M^{\mu-i}) = \frac{\mu(\mu+1)}{2}(\nu - \nu_0)$, while the number of unknowns is $N = \sum_{j<\mu}(b - jw - (\mu - j)\nu_0) + \sum_{\mu \leqslant j \leqslant \lambda}(b - jw) = \sum_{j \leqslant \lambda}(b - jw) - \frac{\mu(\mu+1)}{2}\nu_0$. In other words, we have reduced the number of (linear) unknowns as well as the number of (linear) equations by the same quantity $\frac{\mu(\mu+1)}{2}\nu_0$, which is the number of linear equations used to express the vanishing condition for the ν_0 points $(x_1, 0), \ldots, (x_{\nu_0}, 0)$. (Note that if we were in the more general context of possibly distinct multiplicities, we would have set $y_i = 0$ for the ν_0 points which have the highest multiplicities, in order to maximize the benefit of the re-encoding technique.)

This re-encoding technique is summarized in Algorithm 24. Assuming that Step **4** is done using Algorithm 19 or Algorithm 20, we obtain the following result about the list-decoding of Reed-Solomon codes using this technique.

Corollary 11.7. *Take $r = 1$ and assume the parameters λ, ν, $\mu := \mu_1 = \cdots = \mu_\nu$, b, and $w := w_1$ satisfy $\mathcal{H}_{\text{int},1}$, $\mathcal{H}_{\text{int},2}$, $\mathcal{H}_{\text{int},3}$, and $\mathcal{H}_{\text{int},4}$. Assume further that the points $\{(x_k, y_k)\}_{1 \leqslant k \leqslant \nu}$ satisfy Eq. (11.4) for some $\nu_0 \geqslant w + 1$. Then there exists a probabilistic algorithm that computes a solution to Problem 11 using*

$$\begin{aligned} &\mathcal{O}(\lambda^{\omega-1}\mathsf{M}(\mu^2(\nu - \nu_0)) \log(\nu - \nu_0) + \mu\mathsf{M}(\mu\nu_0) + \mathsf{M}(\nu_0)\log(\nu_0)) \\ \subseteq\ &\tilde{\mathcal{O}}(\lambda^{\omega-1}\mu^2(\nu - \nu_0) + \mu^2\nu_0) \end{aligned}$$

operations in $\mathbb{K}$ with probability of success at least $1/2$.

Proof. For Steps **1**, **2**, and **3**, the complexity analysis is similar to the one in the proof of Proposition 11.4; we still note that we have to compute M_0, so that these steps use $\mathcal{O}(\lambda\mathsf{M}(\mu^2(\nu - \nu_0)) \log(\nu - \nu_0) + \mathsf{M}(\nu_0)\log(\nu_0))$ operations in $\mathbb{K}$. According to Theorem 2.25, Step **4** uses $\mathcal{O}(\lambda^{\omega-1}\mathsf{M}(\mu^2(\nu - \nu_0)) \log(\nu - \nu_0)^2)$ operations in $\mathbb{K}$. Step **5** uses $\mathcal{O}(\mu\mathsf{M}(\mu\nu_0) + \mathsf{M}(\mu^2(\nu - \nu_0)))$ operations in $\mathbb{K}$. Indeed, we first compute $M_0, \ldots, M_0^\mu$ using $\mathcal{O}(\mu\mathsf{M}(\mu\nu_0))$ operations and then the products $M_0^{\mu-j} Q_j$ for $j < \mu$ are computed using $\mathcal{O}(\mu\mathsf{M}(\mu\nu_0) + \mathsf{M}(\mu^2(\nu - \nu_0)))$ operations: for each $j < \mu$, the product $M_0^{\mu-j} Q_j$ can be computed using $\mathcal{O}(\mathsf{M}(\mu\nu_0) + \mathsf{M}(\deg(Q_j)))$ operations since $M_0^{\mu-j}$ has degree at most $\mu\nu_0$;

Algorithm 24 – GurSudIntReenc
(Guruswami-Sudan interpolation step with re-encoding)

Input:

- λ, ν, μ, b, w in $\mathbb{Z}_{>0}$ and satisfying $\mathcal{H}_{\text{int},1}$, $\mathcal{H}_{\text{int},2}$, $\mathcal{H}_{\text{int},3}$, $\mathcal{H}_{\text{int},4}$,
- points $\{(x_k, y_k)\}_{1 \leqslant k \leqslant \nu}$ in $\mathbb{K}^2$ with the x_k pairwise distinct and the y_k satisfying Eq. (11.4).

Output: $Q_0, \ldots, Q_\lambda$ in $\mathbb{K}[X]$ such that $\sum_{j \leqslant \lambda} Q_j Y^j$ is a solution to Problem 11 with input $r = 1$, $\lambda, \nu, \mu = \mu_1 = \cdots = \mu_\nu, b, w$ and $\{(x_k, y_k)\}_{1 \leqslant k \leqslant \nu}$.

1. Compute $n = \mu, m = \lambda + 1, D_i = (\mu - i)(\nu - \nu_0)$, $N_j = b - jw - \nu_0(\mu - j)$ for $j < \mu$ and $N_j = b - jw$ for $\mu \leqslant j \leqslant \lambda$
2. Compute $M_0 = \prod_{1 \leqslant k \leqslant \nu_0}(X - x_k)$ and $\mathfrak{m}_i = \left(\prod_{\nu_0 + 1 \leqslant k \leqslant \nu}(X - x_k)\right)^{\mu - i}$ for $i < \mu$
3. Compute the $F_{i,j}$ for $i < \mu$ and $j \leqslant \lambda$ as in Eq. (11.6)
4. Compute a solution $Q_0, \ldots, Q_\lambda$ to Problem 10 on input n, m, $D_0, \ldots, D_{n-1}$, $N_0, \ldots, N_{m-1}$, and $\{(\mathfrak{m}_i, F_{i,0}, \ldots, F_{i,m-1})\}_{0 \leqslant i < n}$
5. Return $M_0^\mu Q_0, M_0^{\mu-1} Q_1, \ldots, M_0 Q_{\mu-1}, Q_\mu, \ldots, Q_\lambda$, or report "no solution" if Step **4** did

and from Algorithms 19 and 20 we know that $\deg(Q_0) + \cdots + \deg(Q_{\mu-1}) \leqslant (\sum_{i<\mu} D_i) + 1$ (see Eq. (9.1) in Section 9.2), with here $\sum_{i<\mu} D_i = \frac{\mu(\mu+1)}{2}(\nu - \nu_0)$. ☛

Similarly to the remarks following Corollary 11.5, if $\mathrm{Card}(\mathbb{K}) < 24\mu^2(\nu - \nu_0)$ then $\mathbb{K}$ does not contain enough elements to ensure a probability of success at least $1/2$ using our algorithms, but one can solve the problem over an extension of degree $\mathcal{O}(1)$ and retrieve a solution over $\mathbb{K}$ without impacting the cost bound.

11.2.3 Interpolation step in the Wu algorithm

Our goal now is to show that our algorithms can also be used to efficiently solve the interpolation step in the Wu algorithm. In this context, we have $r = 1$ and we make assumptions $\mathcal{H}_{\text{int},1}$, $\mathcal{H}_{\text{int},2}$, and $\mathcal{H}_{\text{int},4}$ on input parameters to Problem 11. We note that here the weight w is no longer related to the dimension of the code; besides, we may have $w \leqslant 0$.

Roughly, the Wu algorithm [Wu08] works as follows. It first uses the Berlekamp-Massey algorithm to reduce the problem of list-decoding a Reed-Solomon code to a problem of rational reconstruction which focuses on the error locations (while the Guruswami-Sudan algorithm directly relies on a problem of polynomial reconstruction which focuses on the correct locations). Then, it solves this problem using an interpolation step and a root-finding step which are very similar to the ones in the Guruswami-Sudan algorithm.

Here we focus on the interpolation step, which differs from the one in the Guruswami-Sudan algorithm by mainly one feature: the points $\{(x_k, y_k)\}_{1\leqslant k\leqslant \nu}$ lie in $\mathbb{K}\times(\mathbb{K}\cup\{\infty\})$, that is, some y_k may take the special value ∞. For a point (x,∞), a polynomial Q in $\mathbb{K}[X,Y]$ and a parameter λ such that $\deg_Y(Q)\leqslant\lambda$, Wu defines in [Wu08] the vanishing condition $Q(x,\infty)=0$ with multiplicity at least μ as the vanishing condition $\overline{Q}(x,0)=0$ with multiplicity at least μ, where $\overline{Q}=Y^\lambda Q(X,Y^{-1})$ is the reversal of Q with respect to the variable Y and the parameter λ. Thus, we have the following direct adaptation of Lemma 11.6.

Lemma 11.8. *Let λ,μ be positive integers, x be an element in $\mathbb{K}$, and $Q=\sum_{j\leqslant\lambda}Q_j(X)Y^j$ be a polynomial in $\mathbb{K}[X,Y]$ with $\deg_Y(Q)\leqslant\lambda$. Then, $Q(x,\infty)=0$ with multiplicity at least μ if and only if $(X-x)^{\mu-j}$ divides $Q_{\lambda-j}$ for each $j<\mu$.*

As in the re-encoding technique, up to reordering the points so that $y_1=\cdots=y_{\nu_\infty}=\infty$ and $y_k\neq\infty$ for $k>\nu_\infty$ for some $\nu_\infty\geqslant 0$, the vanishing condition of Problem 11 restricted to the points $\{(x_k,y_k)\}_{1\leqslant k\leqslant\nu_\infty}$ is equivalent to having $Q_{\lambda-j}=M_\infty^{\mu-j}Q^\star_{\lambda-j}$ for all $j<\mu$, for some unknown polynomials $Q^\star_{\lambda-\mu+1},\ldots,Q^\star_\lambda$ and where $M_\infty=\prod_{1\leqslant k\leqslant\nu_\infty}(X-x_k)$. The degree constraints on $Q_{\lambda-\mu+1},\ldots,Q_\lambda$ directly correspond to degree constraints on $Q^\star_{\lambda-\mu+1},\ldots,Q^\star_\lambda$, while those of $Q_0,\ldots,Q_{\lambda-\mu}$ are unchanged.

This means that in the interpolation problem we are faced with, we can deal with the points of the form (x,∞) the same way we dealt with the points of the form $(x,0)$ in the case of the re-encoding technique: we can pre-solve the corresponding equations efficiently, and we are left with an approximation problem whose dimensions are smaller than if no special attention had been paid when dealing with the points of the form (x,∞). More precisely, let $M=\prod_{\nu_\infty+1\leqslant k\leqslant\nu}(X-x_k)$ and let L be of degree less than $\nu-\nu_\infty$ such that $L(x_k)=y_k$ for each $k>\nu_\infty$. Define further

$$F_{i,j}=\begin{cases}\binom{j}{i}L^{j-i} & \text{for } i\leqslant j\leqslant\lambda-\mu,\\ \binom{j}{i}L^{j-i}M_\infty^{j-\lambda+\mu} & \text{for } \lambda-\mu<j\leqslant\lambda;\end{cases}$$

then we obtain the following simultaneous polynomial approximations: for $i<\mu$,

$$\sum_{i\leqslant j\leqslant\lambda-\mu}F_{i,j}\,Q_j+\sum_{\lambda-\mu<j\leqslant\lambda}F_{i,j}\,Q^\star_j=0 \mod M^{\mu-i}.$$

Pre-solving the equations for the points of the form (x,∞) has led to reduce the number of (linear) unknowns as well as the number of (linear) equations by the same quantity $\frac{\mu(\mu+1)}{2}\nu_\infty$, which is the number of linear equations used to express the vanishing condition for the ν_∞ points $(x_1,\infty),\ldots,(x_{\nu_\infty},\infty)$. We have the following result.

Corollary 11.9. *Take $r=1$ and assume that the parameters λ, ν, $\mu:=\mu_1=\cdots=\mu_\nu$, b, and $w:=w_1$ satisfy $\mathcal{H}_{\mathrm{int},1}$, $\mathcal{H}_{\mathrm{int},2}$, and $\mathcal{H}_{\mathrm{int},4}$. Assume further that each of the points $\{(x_k,y_k)\}_{1\leqslant k\leqslant\nu}$ is allowed to have the special value $y_k=\infty$. Then there exists a probabilistic algorithm that computes a solution to Problem 11 using*

$$\mathcal{O}(\lambda^{\omega-1}\mathsf{M}(\mu^2\nu)\log(\nu))\subseteq\mathcal{O}\tilde{}(\lambda^{\omega-1}\mu^2\nu)$$

operations in $\mathbb{K}$ with probability of success at least $1/2$.

As above, if $\mathrm{Card}(\mathbb{K}) < 24\mu^2(\nu - \nu_\infty)$ then in order to ensure a probability of success at least 1/2 using our algorithms, one can solve the problem over an extension of degree $\mathcal{O}(1)$ and retrieve a solution over $\mathbb{K}$, without impacting the cost bound.

We note that unlike in the re-encoding technique where the focus was on a reduced cost involving $\nu - \nu_0$, here we are not interested in writing the detailed cost involving $\nu - \nu_\infty$. The reason is that ν_∞ is expected to be close to 0 in practice. The main advantage of the Wu algorithm over the Guruswami-Sudan algorithm is that it uses a smaller multiplicity μ, at least for practical code parameters; details about the choice of parameters μ and λ in the context of the Wu algorithm can be found in [BHNW13, Section IV.C].

11.2.4 Slowdown due to repeating points in the soft-decoding

As a last application, we briefly sketch how to adapt our results, and particularly the reduction in Section 11.1, to the context of soft-decoding. Here, we still have $r = 1$; the interpolation step in the soft-decoding of Reed-Solomon codes [KV03a] differs from Problem 11 because there is no assumption ensuring that the x_k are pairwise distinct among the points $\{(x_k, y_k)\}_{1 \leqslant k \leqslant \nu}$. Regarding our algorithms, this is not a minor issue since this assumption is at the core of the reduction in Section 11.1.

We will see that we can still rely on Problem 10 in this context. However, although the number of linear equations $\sum_{1 \leqslant k \leqslant \nu} \frac{\mu_k(\mu_k+1)}{2}$ imposed by the vanishing condition is not changed by the fact that several x_k can be the same field element, the reduction leads to instances of Problem 10 that are solved less efficiently than before. The reason is that there can be significantly more polynomial equations than unknowns in the obtained system of modular equations. In the context of our Algorithms 19 and 20, this means that the displacement rank of the structured matrix may be much larger than if the x_k were pairwise distinct.

To measure to which extent we are far from the situation where the x_k are pairwise distinct, we use the parameter

$$q = \max_{x \in \mathbb{K}} \mathrm{Card}(\{k \in \{1, \ldots, \nu\} \mid x_k = x\}) .$$

For example, $q = 1$ corresponds to pairwise distinct x_k, while $q = \nu$ corresponds to $x_1 = \cdots = x_\nu$; we always have $q \leqslant \nu$ and, if $\mathbb{K}$ is a finite field, $q \leqslant \mathrm{Card}(\mathbb{K})^r$ with $r = 1$ in our context here. Then, we can write the set of points $\mathcal{P} = \{(x_k, y_k)\}_{1 \leqslant k \leqslant \nu}$ as the disjoint union of q sets $\mathcal{P} = \mathcal{P}_1 \cup \cdots \cup \mathcal{P}_q$ where each set $\mathcal{P}_h = \{(x_{h,k}, y_{h,k})\}_{1 \leqslant k \leqslant \nu_h}$ is such that the $x_{h,k}$ are pairwise distinct; we denote $\mu_{h,k}$ the multiplicity associated with the point $(x_{h,k}, y_{h,k})$ in the input of Problem 11. Now, the vanishing condition *(iii)* asks that the q vanishing conditions restricted to each $\mathcal{P}_h$ hold simultaneously. Indeed, $Q(x_k, y_k) = 0$ with multiplicity at least μ_k for all points (x_k, y_k) in $\mathcal{P}$ if and only if for each set $\mathcal{P}_h$, $Q(x_{h,k}, y_{h,k}) = 0$ with multiplicity at least $\mu_{h,k}$ for all points $(x_{h,k}, y_{h,k})$ in $\mathcal{P}_h$.

We have seen in Section 11.1 how to rewrite the vanishing condition as simultaneous polynomial approximations when the x_k are pairwise distinct. This reduction extends to this case: by simultaneously rewriting the vanishing condition for each set $\mathcal{P}_h$, one obtains a problem of simultaneous polynomial approximations whose solutions exactly correspond to the solutions of the instance of (extended) Problem 11 we are considering.

Here, we do not give details about this reduction; they can be found in [Zeh13, Section 5.1.1]. Now, let $\mu^{(h)}$ be the largest multiplicity among those of the points in $\mathcal{P}_h$; in this reduction to Problem 10, the number of polynomial equations we obtain is $\sum_{1\leqslant h\leqslant q}\mu^{(h)}$. Thus, according to Theorem 2.25, for solving this instance of Problem 10, Algorithms 19 and 20 use $\tilde{\mathcal{O}}(\rho^{\omega-1}D)$ operations in $\mathbb{K}$, where $\rho = \max(\lambda+1, \sum_{1\leqslant h\leqslant q}\mu^{(h)})$ and $D = \sum_{1\leqslant k\leqslant \nu}\frac{\mu_r(\mu_r+1)}{2}$. We see in this cost bound that the distribution of the points into disjoint sets $\mathcal{P} = \mathcal{P}_1\cup\cdots\cup\mathcal{P}_q$ has an impact on the number of polynomial equations in the instance of Problem 10 we get: when choosing this distribution, multiplicities could be taken into account in order to minimize this impact.

11.3 The approach based on row reduction

In this section, we summarize the approach for solving Problem 11 via the computation of a reduced polynomial lattice basis; this is a generalization to several variables of the reduction-based approach for the interpolation step of the Coppersmith technique, as presented in Section 10.1. Our main goal here is to give details about our comparison in Section 3.1.2 between the cost bounds for this approach and ours.

Here, $r \geqslant 1$ and for simplicity, we assume that $w := w_1 = \cdots = w_s$ as in the list-decoding of folded Reed-Solomon codes. Besides, we make the assumptions $\mathcal{H}_{\text{int},1}$, $\mathcal{H}_{\text{int},2}$, $\mathcal{H}_{\text{int},3}$, and $\mathcal{H}_{\text{int},4}$ as presented in the introduction. Two main lattice constructions exist in the literature; following [Bus08, §4.5], we present them directly in the case $r \geqslant 1$, and then give the cost bound that can be obtained using polynomial lattice reduction to find a short vector in the lattice.

Let $M = \prod_{1\leqslant k\leqslant\nu}(X - x_k)$ and $L_1, \ldots, L_r \in \mathbb{K}[X]$ such that $\deg(L_j) < \nu$ and $L_j(x_i) = y_{i,j}$ for every $j \in \{1, \ldots, r\}$ and $i \in \{1, \ldots, \nu\}$. In the first construction, the lattice is generated by the polynomials

$$\left\{ M^i \prod_{k=1}^{r}(Y_k - L_k)^{j_k} \,\middle|\, i > 0,\ i + |\boldsymbol{j}| = \mu \right\}$$
$$\bigcup \left\{ \prod_{k=1}^{r}(Y_k - L_k)^{j_k} Y_k^{J_k} \,\middle|\, |\boldsymbol{j}| = \mu,\ |\boldsymbol{J}| \leqslant \lambda - \mu \right\};$$

this construction may be called *banded* due to the shape of these generators when $r = 1$. In the second construction, which may be called *triangular*, the lattice is generated by the polynomials

$$\left\{ M^i \prod_{k=1}^{r}(Y_k - L_k)^{j_k} \,\middle|\, i > 0,\ i + |\boldsymbol{j}| = \mu \right\}$$
$$\bigcup \left\{ \prod_{k=1}^{r}(Y_k - L_k)^{j_k} \,\middle|\, \mu \leqslant |\boldsymbol{j}| \leqslant \lambda \right\}.$$

When $r = 1$, the first construction is used in [BB10, Remark 16] and [LO08, CH15], and the second one is used in [BB10, Ber11]; when $r \geqslant 1$, the former can be found in

[Bus08] while the latter appears in [Bra10, CH12]. In both cases the actual lattice bases are the coefficient vectors (in $\boldsymbol{Y}$) of the polynomials $h(X, X^wY_1, \ldots, X^wY_r)$, for h in either of the sets above; these X^w are introduced to account for the weighted-degree condition *(ii)* in Problem 11.

In this context, for a lattice of dimension m given by generators of degree at most d, the algorithm in [GJV03] computes a shortest vector in the lattice in expected time $\mathcal{O}(m^\omega \mathsf{M}(d)\log(md))$, as detailed below. For a deterministic solution, see the algorithm of Gupta, Sarkar, Storjohann, and Valeriote [GSSV12], whose cost is $\mathcal{O}(m^\omega \mathsf{M}(d)(\log(m)^2 + \log(d)))$.

For the banded basis, its dimension m_B and degree d_B can be taken as follows:

$$m_B = \binom{r+\mu-1}{r} + \binom{r+\mu-1}{r-1}\binom{r+\lambda-\mu}{r}$$

and

$$d_B = \mathcal{O}(\mu\nu).$$

The dimension formula is given explicitly in [Bus08, p. 75], while the degree bound is easily obtained when assuming that the parameters μ, ν, b of Problem 11 satisfy $b \leqslant \mu\nu$; such an assumption is not restrictive, since when $b > \mu\nu$ the polynomial $Q = M^\mu$ is a trivial solution. Then, the arithmetic cost for constructing the lattice matrix with the given generators is $\mathcal{O}\left(\binom{r+\mu}{r}^2 \mathsf{M}(\mu\nu)\right)$, which is $\mathcal{O}(m_B^2\,\mathsf{M}(\mu\nu))$. Similarly, in the triangular case,

$$m_T = \binom{r+\lambda}{r} \quad \text{and} \quad d_T = \mathcal{O}(\lambda\nu),$$

and the cost for constructing the lattice matrix is $\mathcal{O}(m_T^2\,\mathsf{M}(\lambda\nu))$.

Under our assumption $\mathcal{H}_{\text{int},1}$: $\mu \leqslant \lambda$, we always have $m_B \geqslant m_T$ and $d_B \leqslant d_T$; when $r = 1$, we get $m_B = m_T = \lambda + 1$.

To bound the cost of reducing these two polynomial lattice bases, recall that the algorithm of [GJV03] works as follows. Given a basis of a lattice of dimension m and degree d, if $x_0 \in \mathbb{K}$ is given such that the determinant of the lattice does not vanish at $X = x_0$, then the basis will be reduced deterministically using $\mathcal{O}(m^\omega \mathsf{M}(d)\log(md))$ operations in $\mathbb{K}$. Otherwise, such an x_0 is picked at random in $\mathbb{K}$ or, if the cardinality $\mathrm{Card}(\mathbb{K})$ is too small to ensure success with probability at least $1/2$, in a field extension $\mathbb{L}$ of $\mathbb{K}$. In general, $\mathbb{L}$ should be taken of degree $\mathcal{O}(\log(md))$ over $\mathbb{K}$; however, here degree 2 will suffice. Indeed, following [Ber11, p. 206] we note that for the two lattice constructions above the determinants have the special form $M(X)^{i_1}X^{i_2}$ for some $i_1, i_2 \in \mathbb{Z}_{\geqslant 0}$. Since $M(X) = (X - x_1)\cdots(X - x_\nu)$ with $x_1, \ldots, x_\nu \in \mathbb{K}$ pairwise distinct, x_0 can be found deterministically in time $\mathcal{O}(\mathsf{M}(\nu)\log(\nu))$ as soon as $\mathrm{Card}(\mathbb{K}) > \nu + 1$, by evaluating M at $\nu + 2$ arbitrary elements of $\mathbb{K}$; else, $\mathrm{Card}(\mathbb{K})$ is either ν or $\nu + 1$, and x_0 can be found in an extension $\mathbb{L}$ of $\mathbb{K}$ of degree 2. Such an extension can be computed with probability of success at least $1/2$ in time $\mathcal{O}(\log(n))$ (see for example [GG13, Section 14.9]). Then, with the algorithm of [GJV03] we obtain a reduced basis over $\mathbb{L}[X]$ using $\mathcal{O}(m^\omega \mathsf{M}(d)\log(md))$ operations in $\mathbb{L}$; since the degree of $\mathbb{L}$ over $\mathbb{K}$ is $\mathcal{O}(1)$, this is $\mathcal{O}(m^\omega \mathsf{M}(d)\log(md))$ operations in $\mathbb{K}$. Eventually, one can use [SS11, Theorems 13

and 20] to transform this basis into a reduced basis over $\mathbb{K}[X]$ without impacting the cost bound; or more directly, since here we are only looking for a sufficiently short vector in the lattice, this vector can be extracted from a shortest vector in the reduced basis over $\mathbb{L}[X]$. Therefore, by applying the algorithm of [GJV03] to reduce the banded basis and triangular basis shown above, we will always obtain a polynomial Q solution to Problem 11 (assuming one exists) in expected time

$$\mathcal{O}(m_B^\omega \mathsf{M}(\mu\nu)\log(m_B\mu\nu)) \quad \text{and} \quad \mathcal{O}(m_T^\omega \mathsf{M}(\lambda\nu)\log(m_T\lambda\nu)),$$

respectively. In the case $r = 1$, $\mathcal{H}_{\text{int},1}$ implies that these costs are $\mathcal{O}(\lambda^\omega \mathsf{M}(\mu\nu)\log(\lambda\nu))$ and $\mathcal{O}(\lambda^\omega \mathsf{M}(\lambda\nu)\log(\lambda\nu))$, respectively, as reported in [CH11, Ber11]. For $r > 1$, the costs obtained in [Bus08, Bra10] are worse, but only because the short vector algorithms used in those references are slower than the ones we refer to; no cost bound is explicitly given in [CH12]. The result in Corollary 3.1 is an improvement over those of both [Bus08] and [Bra10]. To see this, remark that the cost in our theorem is quasi-linear in $\binom{r+\lambda}{r}^{\omega-1}\binom{r+\mu}{r+1}\nu$, whereas the costs in [Bus08, Bra10] are at least $\binom{r+\lambda}{r}^{\omega}\mu\nu$; a simplification proves our claim.

11.4 On assumption $\mathcal{H}_{\text{int},1}$

In this section, we discuss the relevance of the assumption $\mathcal{H}_{\text{int},1}$ introduced previously for Problem 11. In the introduction, we did not make any assumption on $\mu = \max_{1\leqslant i\leqslant \nu} \mu_i$ and λ, but we mentioned that the assumption $\mathcal{H}_{\text{int},1}$, that is, $\mu \leqslant \lambda$ is mostly harmless. The following lemma substantiates this claim, by showing that the case $\mu > \lambda$ can be reduced to the case $\mu = \lambda$.

Lemma 11.10. *Let $r, \lambda, \nu, \mu_1, \ldots, \mu_\nu, b, \boldsymbol{w}$ be parameters for Problem 11, and suppose that $\mu > \lambda$. Define $P = \prod_{1\leqslant i\leqslant \nu:\ \mu_i>\lambda}(X - x_i)^{\mu_i-\lambda}$ and $d = \deg(P)$. The solutions to this problem are the polynomials of the form $Q = Q^\star P$ with $Q^\star$ a solution for the parameters $r, \lambda, \nu, \mu'_1, \ldots, \mu'_\nu, b - d, \boldsymbol{w}$, where $\mu'_i = \lambda$ if $\mu_i > \lambda$ and $\mu'_i = \mu_i$ otherwise.*

Proof. Assume a solution exists, say Q, and let $Q_i(X, \boldsymbol{Y}) = Q(X+x_i, Y_1+y_{i,1}, \ldots, Y_r+y_{i,r})$ for $i = 1, \ldots, \nu$. Every monomial of Q_i has the form $X^h \boldsymbol{Y}^{\boldsymbol{j}}$ with $h \geqslant \mu_i - \lambda$, since $|\boldsymbol{j}| \leqslant \lambda$ by condition *(i)* and $h + |\boldsymbol{j}| \geqslant \mu_i$ by condition *(iii)*. Therefore, if $\mu_i > \lambda$ then $X^{\mu_i-\lambda}$ divides Q_i and, shifting back the coordinates for each i, we deduce that P divides Q.

Let us now consider the polynomial $Q^\star = Q/P$ and show that it solves Problem 11 for the parameters $r, \lambda, \nu, \mu'_1, \ldots, \mu'_\nu, b-d, \boldsymbol{w}$. First, $Q^\star$ clearly satisfies conditions *(i)* and *(i)*. Furthermore, writing $Q = \sum_{\boldsymbol{j}} Q_{\boldsymbol{j}}(X)\boldsymbol{Y}^{\boldsymbol{j}}$ and $Q^\star = \sum_{\boldsymbol{j}} Q^\star_{\boldsymbol{j}}(X)\boldsymbol{Y}^{\boldsymbol{j}}$, we have $Q^\star_{\boldsymbol{j}} = Q_{\boldsymbol{j}}/P$ for all $\boldsymbol{j}$, so that

$$\begin{aligned}\operatorname{wdeg}_{\boldsymbol{w}}(Q^\star) &= \max_{\boldsymbol{j}}(\deg(Q_{\boldsymbol{j}}) - d + w_1 j_1 + \cdots + w_r j_r)\\ &= \operatorname{wdeg}_{\boldsymbol{w}}(Q) - d\\ &< b - d,\end{aligned}$$

so that condition *(ii)* holds for $Q^\star$ with b replaced by $b-d$. Finally, $Q^\star$ satisfies condition *(iii)* with each $\mu_i > \lambda$ replaced by $\mu'_i = \lambda$: writing $Q^\star_i(X, \boldsymbol{Y}) = Q^\star(X+x_i, Y_1+y_{i,1}, \ldots, Y_r+$

$y_{i,r}$) for $i \in \{1, \dots, \nu\}$ such that $\mu_i > \lambda$, we have

$$Q_i^\star(X, \boldsymbol{Y}) = \frac{Q_i(X, \boldsymbol{Y})}{X^{\mu_i - \lambda} P_i(X)},$$

where

$$P_i(X) = \prod_{h \neq i:\ \mu_h > \lambda} (X + x_i - x_h)^{\mu_h - \lambda}.$$

All the monomials of $Q_i(X, \boldsymbol{Y})/X^{\mu_i-\lambda}$ have the form $X^h \boldsymbol{Y}^{\boldsymbol{j}}$ with $h + |\boldsymbol{j}| \geqslant \mu_i - (\mu_i - \lambda) = \lambda$ and, since $P_i(0) \neq 0$, the same holds for $Q_i^\star(X, \boldsymbol{Y})$.

Conversely, let Q' be *any* solution to Problem 11 with parameters $r, \lambda, \nu, \mu'_1, \dots, \mu'_\nu, b - d, \boldsymbol{w}$. Proceeding as in the previous paragraph, one easily verifies that the product $Q' P$ is a solution to Problem 11 with parameters $r, \lambda, \nu, \mu_1, \dots, \mu_\nu, b, \boldsymbol{w}$. ☙

11.5 On assumption $\mathcal{H}_{\mathrm{int},3}$

In this section, we show the relevance of the assumption "$w_j < \nu$ for some $j \in \{1, \dots, r\}$" when considering Problem 11; in particular when $r = 1$ or when we assume that $w_1 = \cdots = w_r =: w$, this shows the relevance of the assumption $\mathcal{H}_{\mathrm{int},3}$: $0 \leqslant w < \nu$. More precisely, when $w_j \geqslant \nu$ for every j, Lemma 11.11 below gives an explicit solution to Problem 11.

Lemma 11.11. *Let $r, \lambda, \nu, \boldsymbol{\mu}, b, \boldsymbol{w}$ be parameters for Problem 11 and suppose that $w_j \geqslant \nu$ for all $j \in \{1, \dots, r\}$. Define $P = \prod_{1 \leqslant i \leqslant \nu} (X - x_i)^{\mu_i}$ and $d = \deg(P) = \sum_{1 \leqslant i \leqslant \nu} \mu_i$. If $b \leqslant d$ then this problem has no solution. Otherwise, a solution is given by the polynomial P, considered as an element of $\mathbb{K}[X, \boldsymbol{Y}]$.*

Proof. If $b > d$ then it is easily checked that P satisfies conditions *(i)–(iii)* and thus solves Problem 11. Now, let us show that if Problem 11 admits a solution Q, then $b > d$ must hold. Let $d_Y = \deg_{\boldsymbol{Y}} Q$. If $d_Y \geqslant \mu = \max_i \mu_i$, then the weighted-degree condition *(ii)* gives $b > \mathrm{wdeg}_{\boldsymbol{w}}(Q) \geqslant d_Y(\min_j w_j) \geqslant \mu\nu \geqslant d$. Let us then assume $d_Y < \mu$. Following the proof of Lemma 11.10, we can write $Q = P^\star Q^\star$ where $P^\star = \prod_{1 \leqslant i \leqslant \nu:\ \mu_i > d_Y} (X - x_i)^{\mu_i - d_Y}$, for some $Q^\star$ in $\mathbb{K}[X, \boldsymbol{Y}]$ such that $\deg_{\boldsymbol{Y}} Q^\star = d_Y$. Then, the weighted-degree condition gives $b > \sum_{1 \leqslant i \leqslant \nu:\ \mu_i > d_Y} (\mu_i - d_Y) + \mathrm{wdeg}_{\boldsymbol{w}}(Q^\star) \geqslant \sum_{1 \leqslant i \leqslant \nu:\ \mu_i > d_Y} (\mu_i - d_Y) + d_Y \nu \geqslant \sum_{1 \leqslant i \leqslant \nu:\ \mu_i > d_Y} \mu_i + \sum_{1 \leqslant i \leqslant \nu:\ \mu_i \leqslant d_Y} d_Y \geqslant d$. ☙

12

Some tools for computing with polynomial matrices

In this section, we present some tools that will be used in algorithms for interpolant bases. In particular, we give a detailed cost analysis with logarithmic factors for the algorithms in [ZLS12] for computing products of bases with unbalanced row degrees and minimal kernel bases. To derive cost bounds, it will be convenient to use additional time functions.

12.1 More time functions for polynomial matrices

In Section 6.1, we defined time functions $\mathsf{MM}(m, d)$ and $\mathsf{MM}''(m, d)$ related to multiplication of polynomial matrices and divide-and-conquer computations splitting some degree in two halves. In the next chapters, to simplify the cost analyses it will be convenient to have at hand additional such functions, that we define now and which typically arise when dealing with matrices that have non-uniform degrees and in divide-and-conquer computations involving these. We remark that similar functions were already used, for example in [Sto03, GJV03].

Definition 12.1. *Let m and d be two positive real numbers. Then, we define*

- $\mathsf{MM}'(m, d) = \sum_{0 \leqslant i \leqslant \log(m)} 2^i \mathsf{MM}(2^{-i}m, 2^i d)$,
- $\overline{\mathsf{MM}'}(m, d) = \sum_{0 \leqslant i \leqslant \log(m)} 2^i \mathsf{MM}'(2^{-i}m, 2^i d)$,
- $\overline{\mathsf{MM}''}(m, d) = \sum_{0 \leqslant i \leqslant \log(m)} 2^i \mathsf{MM}''(2^{-i}m, 2^i d)$.

We now give some upper bounds for these quantities, since once the cost bounds are obtained using these functions, it will also be convenient to know how these can be estimated. The reader not interested in these technical details may of course skip this section. (For similar bounds in the case $\omega = 2$, we refer to [JNSV17, Appendix A].)

Lemma 12.2. *We have the upper bound*

$$\mathsf{MM}'(m, d) \in \mathcal{O}(m^{\omega-1}\mathsf{M}(md)).$$

Proof. It is enough to show this bound for m and d powers of 2. Using the super-linearity $\mathcal{H}_{\mathsf{M}(\cdot)}$ we obtain $\mathsf{M}(2^{-i}md) \leqslant 2^{-i}\mathsf{M}(md)$, hence

$$\mathsf{MM}'(m,d) = \sum_{0\leqslant i\leqslant \log(m)} 2^{-i} m \mathsf{MM}(2^i, 2^{-i}md) \in \mathcal{O}\Big(\sum_{0\leqslant i\leqslant \log(m)} 2^{i(\omega-2)} m\mathsf{M}(md)\Big).$$

This implies the result since, using $\omega > 2$, we have $\sum_{0\leqslant i\leqslant \log(m)} 2^{i(\omega-2)} \in \Theta(m^{\omega-2})$. ☙

Lemma 12.3. *We have the upper bounds*

$$\begin{aligned}\overline{\mathsf{MM}'}(m,d) &\in \mathcal{O}(m^{\omega-1}\mathsf{M}(md)),\\ \overline{\mathsf{MM}''}(m,d) &\in \mathcal{O}(m^{\omega-1}\mathsf{M}(md) + m^{\omega}\mathsf{M}(d)\log(d)).\end{aligned}$$

Proof. It is enough to show these bounds for m and d powers of 2.

The first one is obtained from Lemma 12.2, which implies that

$$\sum_{i=0}^{\log(m)} 2^i \mathsf{MM}'(2^{-i}m, 2^i d) \in \mathcal{O}\left(\sum_{i=0}^{\log(m)} 2^{i(2-\omega)} m^{\omega-1}\mathsf{M}(md)\right),$$

and from the fact that $\sum_{0\leqslant i\leqslant \log(m)} 2^{i(2-\omega)}$ is upper bounded by a constant since $\omega > 2$.

Now, we focus on the second bound. By definition,

$$\overline{\mathsf{MM}''}(m,d) \in \mathcal{O}\left(\sum_{i=0}^{\log(m)} 2^i \sum_{j=0}^{\log(2^i d)} 2^j (2^{-i}m)^{\omega}\mathsf{M}(2^{i-j}d)\right).$$

In the inner sum, j goes from 0 to $\log(2^i d) = i + \log(d)$: we will separately study the first terms with $j \leqslant i$ and the remaining terms with $j > i$.

First, using the super-linearity property $\mathsf{M}(2^{i-j}d) \leqslant 2^{i-j}\mathsf{M}(md)/m$, we obtain

$$\begin{aligned}\sum_{i=0}^{\log(m)} 2^i \sum_{j=0}^{i} 2^j (2^{-i}m)^{\omega}\mathsf{M}(2^{i-j}d) &\in \mathcal{O}\left(\sum_{i=0}^{\log(m)} (i+1) 2^{i(2-\omega)} m^{\omega-1}\mathsf{M}(md)\right)\\ &\in \mathcal{O}(m^{\omega-1}\mathsf{M}(md)),\end{aligned}$$

since the sum $\sum_{0\leqslant i\leqslant \log(m)} (i+1)2^{i(2-\omega)}$ is less than its limit $(1-2^{2-\omega})^{-2}$ when $m \to \infty$.

Then, using the super-linearity property $\mathsf{M}(2^{i-j}d) \leqslant 2^{i-j}\mathsf{M}(d)$ when $j > i$, we obtain

$$\begin{aligned}\sum_{i=0}^{\log(m)} 2^i \sum_{j=i+1}^{i+\log(d)} 2^j (2^{-i}m)^{\omega}\mathsf{M}(2^{i-j}d) &\leqslant \sum_{i=0}^{\log(m)} 2^{i(2-\omega)} m^{\omega}\mathsf{M}(d)\log(d)\\ &\in \mathcal{O}(m^{\omega}\mathsf{M}(d)\log(d)),\end{aligned}$$

which concludes the proof. ☙

In some cost analyses, we will also encounter the following quantities.

Lemma 12.4. *We have the upper bounds*

$$\begin{aligned}\textstyle\sum_{0\leqslant i\leqslant \log(d)} 2^i\overline{\mathsf{MM}'}(m,2^{-i}d) &\in \mathcal{O}(m^{\omega-1}\mathsf{M}(md)+m^{\omega}\mathsf{M}(d)\log(d)),\\ \textstyle\sum_{0\leqslant i\leqslant \log(d)} 2^i\overline{\mathsf{MM}''}(m,2^{-i}d) &\in \mathcal{O}(m^{\omega-1}\mathsf{M}(md)+m^{\omega}\mathsf{M}(d)\log(d)^2).\end{aligned}$$

Proof. It is enough to show these bounds for m and d powers of 2.

Let us study the first bound. By definition,

$$\begin{aligned}\sum_{i=0}^{\log(d)} 2^i\overline{\mathsf{MM}'}(m,2^{-i}d) &= \sum_{i=0}^{\log(d)} 2^i \sum_{j=0}^{\log(m)} 2^j \sum_{k=0}^{\log(m)-j} 2^k\mathsf{MM}(2^{-j-k}m,2^{j+k-i}d)\\ &= \sum_{j=0}^{\log(m)}\sum_{k=0}^{\log(m)-j}\sum_{i=0}^{\log(d)} 2^{i+j+k}\mathsf{MM}(2^{-j-k}m,2^{j+k-i}d).\end{aligned}$$

Considering the terms with $i\leqslant j+k$, we use $\mathsf{M}(2^{j+k-i}d)\leqslant 2^{j+k-i}\mathsf{M}(md)/m$ to obtain

$$\begin{aligned}&\sum_{j=0}^{\log(m)}\sum_{k=0}^{\log(m)-j}\sum_{i=0}^{j+k}2^{i+j+k}\mathsf{MM}(2^{-j-k}m,2^{j+k-i}d)\\ &\qquad\in \mathcal{O}\left(\sum_{j=0}^{\log(m)}\sum_{k=0}^{\log(m)-j}(j+k+1)2^{(j+k)(2-\omega)}m^{\omega-1}\mathsf{M}(md)\right),\end{aligned}$$

from which we conclude, since the sum $\sum_{0\leqslant j\leqslant\log(m)}\sum_{0\leqslant k\leqslant\log(m)-j}(j+k+1)2^{(j+k)(2-\omega)}$ is bounded by a constant.

Now, considering the terms with $i>j+k$, we use $\mathsf{M}(2^{j+k-i}d)\leqslant 2^{j+k-i}\mathsf{M}(d)$ to obtain

$$\begin{aligned}&\sum_{j=0}^{\log(m)}\sum_{k=0}^{\log(m)-j}\sum_{i=j+k+1}^{\log(d)}2^{i+j+k}\mathsf{MM}(2^{-j-k}m,2^{j+k-i}d)\\ &\qquad\in \mathcal{O}\left(\sum_{j=0}^{\log(m)}\sum_{k=0}^{\log(m)-j}2^{(j+k)(2-\omega)}m^{\omega}\mathsf{M}(d)\log(d)\right),\end{aligned}$$

where again the sum on j and k is $\mathcal{O}(1)$ since $\omega>2$.

Then, we study the second bound. By definition,

$$\begin{aligned}\sum_{i=0}^{\log(d)} 2^i\overline{\mathsf{MM}''}(m,2^{-i}d) &= \sum_{i=0}^{\log(d)} 2^i \sum_{j=0}^{\log(m)} 2^j \sum_{k=0}^{\log(d)+j-i} 2^k\mathsf{MM}(2^{-j}m,2^{j-i-k}d)\\ &= \sum_{j=0}^{\log(m)}\sum_{i=0}^{\log(d)}\sum_{k=0}^{\log(d)+j-i} 2^{i+j+k}\mathsf{MM}(2^{-j}m,2^{j-i-k}d).\end{aligned}$$

Considering the terms with $k>j-i$, we use $\mathsf{M}(2^{j-i-k}d)\leqslant 2^{j-i-k}\mathsf{M}(d)$ to obtain

$$\sum_{j=0}^{\log(m)}\sum_{i=0}^{\log(d)}\sum_{k=\max(0,1+j-i)}^{\log(d)+j-i} 2^{i+j+k}\mathsf{MM}(2^{-j}m,2^{j-i-k}d)\in\mathcal{O}\left(\sum_{j=0}^{\log(m)}2^{j(2-\omega)}m^{\omega}\mathsf{M}(d)\log(d)^2\right);$$

this is $\mathcal{O}(m^\omega \mathsf{M}(d)\log(d)^2)$ since $\omega > 2$.

Now, considering the terms with $0 \leqslant k \leqslant j-i$, and thus also $i \leqslant j$, the super-linearity property yields $\mathsf{M}(2^{j-i-k}d) \leqslant 2^{j-i-k}\mathsf{M}(md)/m$ and therefore

$$\sum_{j=0}^{\log(m)} \sum_{i=0}^{j} \sum_{k=0}^{j-i} 2^{i+j+k}\mathsf{MM}(2^{-j}m, 2^{j-i-k}d) \in \mathcal{O}\left(\sum_{j=0}^{\log(m)} (j+1)^2 2^{j(2-\omega)} m^{\omega-1}\mathsf{M}(md)\right).$$

This gives the conclusion, since $\omega > 2$ implies that $\sum_{j=0}^{\log(m)}(j+1)^2 2^{j(2-\omega)}$ is in $\mathcal{O}(1)$. ☙

12.2 Multiplying matrices with unbalanced row degrees

In this section, we give a detailed complexity analysis concerning the fast algorithm from [ZLS12, Section 3.6] for the multiplication of matrices with controlled, yet possibly unbalanced, row degrees. For the sake of exposition, it is recalled as Algorithm 25. It will be a central building block in our algorithm for interpolant bases (Algorithm 28), which computes the sought basis as the product of two interpolant bases obtained recursively. It is also used for the multiplication of kernel bases that occur in the algorithm of [ZLS12], which we will rely on to perform a change of shift.

In order to multiply matrices with unbalanced row degrees, we use a technique based on *partial linearization*, similarly to ideas used in Section 6.3. This can be seen as a simplified version of the one in [GSSV12, Section 6] for the purpose of multiplication. For a matrix $\mathbf{B}$ with sum of row degrees ξ, meant to be the left operand in a product $\mathbf{BA}$ for some $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$, this technique consists in expanding the high-degree rows of $\mathbf{B}$ so as to obtain a matrix $\widetilde{\mathbf{B}}$ with $\mathcal{O}(m)$ rows and degree at most ξ/m, then computing the product $\widetilde{\mathbf{B}}\mathbf{A}$, and finally retrieving the actual product $\mathbf{BA}$ by grouping together the rows that have been expanded (this inverse operation is called *partial compression* in what follows).

More precisely, let $\mathbf{B} \in \mathbb{K}[X]^{k\times m}$ for some k and m, with $\operatorname{rdeg}(\mathbf{B}) = (d_1, \dots, d_k)$ and write $\xi = d_1 + \cdots + d_k$. We are given a target degree bound d. For each i, the row $\mathbf{B}_{i,*}$ of degree d_i is expanded into $\alpha_i = 1 + \lfloor d_i/(d+1)\rfloor$ rows $\widetilde{\mathbf{B}}_{(i,0),*}, \dots, \widetilde{\mathbf{B}}_{(i,\alpha_i-1),*}$ of degree at most d, related by the identity

$$\mathbf{B}_{i,*} = \widetilde{\mathbf{B}}_{(i,0),*} + X^{d+1}\widetilde{\mathbf{B}}_{(i,1),*} + \cdots + X^{(\alpha_i-1)(d+1)}\widetilde{\mathbf{B}}_{(i,\alpha_i-1),*}. \tag{12.1}$$

Then, the expanded matrix $\widetilde{\mathbf{B}}$ has $\sum_{1\leqslant i\leqslant k}\alpha_i \leqslant k + \xi/(d+1)$ rows $\widetilde{\mathbf{B}}_{(i,j),*}$. We will mainly use this technique for $k \leqslant m$ and $d = \lfloor \xi/m \rfloor$ or $d = \lceil \xi/m \rceil$, in which case $\widetilde{\mathbf{B}}$ has fewer than $2m$ rows. The partial compression is the computation of the row i of the product $\mathbf{BA}$ from the rows $(i,0), \dots, (i, \alpha_i - 1)$ of $\widetilde{\mathbf{B}}\mathbf{A}$ using the formula in Eq. (12.1).

Proposition 12.5. *Let $\mathbf{A}$ and $\mathbf{B}$ in $\mathbb{K}[X]^{m\times m}$, and $\mathbf{d} = \operatorname{rdeg}(\mathbf{A})$. Let $\xi \in \mathbb{Z}_{>0}$ be such that $\xi \geqslant m$, $|\mathbf{d}| \leqslant \xi$, and $|\operatorname{rdeg}_{\mathbf{d}}(\mathbf{B})| \leqslant \xi$. Then, the product $\mathbf{BA}$ can be computed using*

$$\mathcal{O}(\mathsf{MM}'(m, \xi/m)) \subseteq \mathcal{O}(m^{\omega-1}\mathsf{M}(\xi))$$

operations in $\mathbb{K}$.

Proof. In this proof, we use notation from Algorithm 25. The correctness of this algorithm follows from the identity $\mathbf{BA} = \hat{\mathbf{B}}\hat{\mathbf{A}} = \mathbf{B}_0\mathbf{A}_0 + \mathbf{B}_1\mathbf{A}_1 + \cdots + \mathbf{B}_\ell\mathbf{A}_\ell$. In what follows we focus on proving the cost bound $\mathcal{O}(\mathsf{MM}'(m, \xi/m))$; the announced upper bounds on this quantity follow from Equation (6.3) and Lemma 12.2.

We start with Step **6**, which adds the $m \times m$ matrices $\mathbf{P}_i = \mathbf{B}_i\mathbf{A}_i$ obtained after the first five steps. For each i in $\{0, \ldots, \ell\}$, we have $\mathrm{rdeg}(\mathbf{P}_i) \leqslant \mathrm{rdeg}(\mathbf{BA}) \leqslant \mathrm{rdeg}_{\mathbf{d}}(\mathbf{B})$ componentwise, hence $|\mathrm{rdeg}(\mathbf{P}_i)| \leqslant \xi$. Recalling that $\ell = \lceil \log(m) \rceil$, the sum at Step **6** thus uses $\mathcal{O}(m\xi\ell) = \mathcal{O}(m\xi \log(m))$ additions in $\mathbb{K}$. On the other hand, by definition of $\mathsf{MM}'(\cdot, \cdot)$, the trivial lower bound $\mathsf{MM}(n, d) \geqslant n^2 d$ for any n, d implies that $m\xi \log(m) \in \mathcal{O}(\mathsf{MM}'(m, \xi/m))$.

Now we study the for loop. We remark that only Step **5.c** involves arithmetic operations in $\mathbb{K}$. Therefore the main task is to give bounds on the dimensions and degrees of the matrices we multiply at Step **5.c**. For i in $\{0, \ldots, \ell\}$, the column dimension of $\mathbf{A}_i$ is m and the row dimension of $\mathbf{B}_i^{\emptyset}$ is k_i. We further denote by m_i the row dimension of $\mathbf{A}_i$ (and column dimension of $\widetilde{\mathbf{B}}_i^{\emptyset}$), and we write $[d_{\pi(1)}, \ldots, d_{\pi(m)}] = [\mathbf{d}_0 | \cdots | \mathbf{d}_\ell]$ where the sizes of $\mathbf{d}_0, \ldots, \mathbf{d}_\ell$ correspond to those of the blocks of $\hat{\mathbf{B}}$ as in Step **4** of the algorithm.

First, let $i = 0$. Then, $\mathbf{A}_0$ is $m_0 \times m$ of degree at most ξ/m and $\mathbf{B}_0$ is $k_0 \times m_0$ with $m_0 \leqslant m$ and $k_0 \leqslant m$ (we note that for $i = 0$ these may be equalities and thus one does not need to discard the zero rows of $\mathbf{B}_0$ to obtain efficiency). Besides, we have the componentwise inequality $\mathrm{rdeg}(\mathbf{B}_0) \leqslant \mathrm{rdeg}_{\mathbf{d}_0}(\mathbf{B}_0) \leqslant \mathrm{rdeg}_{\mathbf{d}}(\mathbf{B})$, so that $|\mathrm{rdeg}(\mathbf{B}_0)| \leqslant |\mathrm{rdeg}_{\mathbf{d}}(\mathbf{B})| \leqslant \xi$. Then, $\mathbf{B}_0^{\emptyset}$ can be partially linearized into a matrix $\widetilde{\mathbf{B}}_0^{\emptyset}$ which has at most $2m$ rows and degree at most ξ/m, and the computation at Step **5.c** for $i = 0$ uses $\mathcal{O}(\mathsf{MM}(m, \xi/m))$ operations.

Now, let $i \in \{1, \ldots, \ell\}$. By assumption, the sum of the row degrees of $\mathbf{A}$ does not exceed ξ: since all rows in $\mathbf{A}_i$ have degree more than $2^{i-1}\xi/m$, this implies that $m_i < m/2^{i-1}$. Besides, since $\min(\mathbf{d}_i) > 2^{i-1}\xi/m$, we obtain that every nonzero row of $\mathbf{B}_i$ has $\mathbf{d}_i$-row degree more than $2^{i-1}\xi/m$. Then, $\xi \geqslant |\mathrm{rdeg}_{\mathbf{d}}(\mathbf{B})| \geqslant |\mathrm{rdeg}_{\mathbf{d}_i}(\mathbf{B}_i^{\emptyset})| > k_i 2^{i-1}\xi/m$ implies that $k_i < m/2^{i-1}$. Furthermore, since we have $|\mathrm{rdeg}(\mathbf{B}_i^{\emptyset})| = |\mathrm{rdeg}(\mathbf{B}_i)| \leqslant \xi$, the partial linearization at Step **5.b** can be done by at most doubling the number of rows of $\mathbf{B}_i^{\emptyset}$, producing $\widetilde{\mathbf{B}}_i^{\emptyset}$ with fewer than $2m/2^{i-1}$ rows and of degree at most $2^{i-1}\xi/m$. To summarize: $\mathbf{A}_i$ has m columns, $m_i < m/2^{i-1}$ rows, and degree at most $2^i\xi/m$; $\widetilde{\mathbf{B}}_i^{\emptyset}$ has fewer than $2m/2^{i-1}$ rows, and degree less than $2^i\xi/m$. Then, the computation of $\widetilde{\mathbf{P}}_i^{\emptyset}$ uses $\mathcal{O}(2^i\mathsf{MM}(m/2^{i-1}, 2^i\xi/m))$ operations in $\mathbb{K}$. Thus, overall the for loop uses $\mathcal{O}(\mathsf{MM}'(m, \xi/m))$ operations in $\mathbb{K}$. ☙

12.3 Detailed cost bound for the kernel basis algorithm of [ZLS12]

Here, we give a detailed cost analysis for the minimal kernel basis algorithm in [ZLS12]. We rewrite it in Algorithm 26, using our convention here that basis vectors are rows of the basis matrix (whereas in the above reference they are its columns), and furthermore assuming that the input matrix has full rank. This simplifies the cost analysis by allowing

Algorithm 25 – RDegPolMatMul
(Multiplication with unbalanced row degrees [ZLS12])

Input:

- polynomial matrices $\mathbf{A}$ and $\mathbf{B}$ in $\mathbb{K}[X]^{m\times m}$,
- an integer ξ with $\xi \geqslant m$, $|\mathrm{rdeg}(\mathbf{A})| \leqslant \xi$, and $|\mathrm{rdeg}_{\mathbf{d}}(\mathbf{B})| \leqslant \xi$.

Output: the product $\mathbf{P} = \mathbf{BA}$.

1. $(d_1, \ldots, d_m) \leftarrow \mathrm{rdeg}(\mathbf{A})$
2. $\pi \leftarrow$ a permutation of $\{1, \ldots, m\}$ such that $(d_{\pi(1)}, \ldots, d_{\pi(m)})$ is non-decreasing
3. $\hat{\mathbf{A}} \leftarrow \pi\mathbf{A}$ and $\hat{\mathbf{B}} \leftarrow \mathbf{B}\pi^{-1}$
4. define $\ell = \lceil \log(m) \rceil$ and the row blocks $\hat{\mathbf{A}} = [\mathbf{A}_0^\mathsf{T} \quad \mathbf{A}_1^\mathsf{T} \quad \cdots \quad \mathbf{A}_\ell^\mathsf{T}]^\mathsf{T}$, where the rows in $\mathbf{A}_0$ have row degree at most ξ/m and for $i = 1, \ldots, \ell$ the rows in $\mathbf{A}_i$ have row degree in $\{2^{i-1}\xi/m + 1, \ldots, 2^i\xi/m\}$
5. define $\hat{\mathbf{B}} = [\mathbf{B}_0 \;\; \mathbf{B}_1 \;\; \cdots \;\; \mathbf{B}_\ell]$ the corresponding column blocks of $\hat{\mathbf{B}}$
6. For i from 0 to ℓ:
   - **a.** Read $r_1, \ldots, r_{k_i}$ the indices of the nonzero rows in $\mathbf{B}_i$ and define $\mathbf{B}_i^\emptyset$ the submatrix of $\mathbf{B}_i$ obtained by removing the zero rows
   - **b.** $\widetilde{\mathbf{B}}_i^\emptyset \leftarrow$ partial linearization of $\mathbf{B}_i^\emptyset$ with $\deg(\widetilde{\mathbf{B}}_i^\emptyset) \leqslant 2^i\xi/m$
   - **c.** $\widetilde{\mathbf{P}}_i^\emptyset \leftarrow \widetilde{\mathbf{B}}_i^\emptyset \mathbf{A}_i$
   - **d.** Perform the partial compression $\mathbf{P}_i^\emptyset \leftarrow \widetilde{\mathbf{P}}_i^\emptyset$
   - **e.** Re-introduce the zero rows to obtain $\mathbf{P}_i$, which is $\mathbf{B}_i\mathbf{A}_i$ (its rows at indices $r_1, \ldots, r_{k_i}$ are those of $\mathbf{P}_i^\emptyset$, its other rows are zero)
7. Return $\mathbf{P} = \mathbf{P}_0 + \mathbf{P}_1 + \cdots + \mathbf{P}_\ell$

us to better control the dimensions of the matrices encountered in the computations: in the recursive calls, we always have input matrices with more rows than columns.

Remark 12.6. We defined kernel bases in Definition 8.1. In [ZLS12], the same bases were called *nullspace* bases; however, the authors later switched to *kernel* basis in [ZL13]. They explain this change by the desire to avoid possible ambiguity: the term "nullspace" is more commonly used for matrices over a field, and would thus refer to nullspace elements that form a vector space over the fractions $\mathbb{K}(X)$, while here we are interested in kernel elements that form a module over the polynomials $\mathbb{K}[X]$. ☕

We have seen in Section 7.1 that the quantity $\mathsf{MM}''(m,d) = \sum_{0\leqslant j\leqslant \log(d)} 2^j \mathsf{MM}(m, 2^{-j}d)$ from Definition 6.1 arises in the cost analysis of fast algorithms for the computation of approximant bases [BL94, GJV03], whose divide-and-conquer strategy consists in finding two subproblems which split the order d in two halves. The kernel basis algorithm in [ZLS12] follows a divide-and-conquer approach on the dimension of the input matrix, and computes at each node of the recursion some products of matrices with unbalanced row degrees as well as a minimal approximant basis. In particular, its cost will be expressed using the quantities $\overline{\mathsf{MM}'}(m,d)$ and $\overline{\mathsf{MM}''}(m,d)$ introduced in Definition 12.1.

The following result refines the cost analysis in [ZLS12, Theorem 4.1], counting the logarithmic factors.

Proposition 12.7. *Algorithm 26 is correct. Let us assume that $m \in \mathcal{O}(n)$, and let ξ be an integer such that $\xi \geqslant m$ and $|\mathbf{s}| \leqslant \xi$. Then, Algorithm 26 uses*

$$\begin{aligned}&\mathcal{O}(\overline{\mathsf{MM}'}(m,\xi/m) + \overline{\mathsf{MM}''}(m,\xi/m))\\ &\quad\subseteq\ \mathcal{O}(m^{\omega-1}\mathsf{M}(\xi) + m^{\omega}\mathsf{M}(\xi/m)\log(\xi/m))\end{aligned}$$

operations in $\mathbb{K}$.

Proof. The proof of correctness can be found in [ZLS12]. Here, we prove the cost bound following the algorithm step by step.

Step **1**: since $\rho \leqslant |\mathbf{s}| \leqslant \xi$, we have $\lambda \leqslant \lceil \xi/n \rceil$.

Step **2**: according to Proposition 7.4 (or [GJV03, Theorem 2.4]), $\mathbf{P}$ can be computed using $\mathcal{O}(\mathsf{MM}''(m,\lambda))$ operations in $\mathbb{K}$. Since $\lambda \leqslant \lceil \xi/n \rceil$ and $m \in \mathcal{O}(n)$, this step uses $\mathcal{O}(\mathsf{MM}''(n,\xi/n))$ operations. Besides, from Theorem 1.11 and Lemma 2.10, we have $|\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})| = \deg(\det(\mathbf{P})) + |\mathbf{s}| \leqslant 3n\lambda + \xi \leqslant 3(\rho + m) + \xi \leqslant 7\xi$.

Step **3**: finding $\mathbf{P}_1$ and $\mathbf{P}_2$ can be done by computing $\mathbf{PF}$. The matrix $\mathbf{F}$ is $m \times n$ with row degree $\mathbf{w} = \mathrm{rdeg}(\mathbf{F}) \leqslant \mathbf{s}$; in particular, $|\mathbf{w}| \leqslant \xi$. Besides, $\mathbf{P}$ is an $m \times m$ matrix and $|\mathrm{rdeg}_{\mathbf{w}}(\mathbf{P})| \leqslant |\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})| \leqslant 7\xi$. Then, one can augment $\mathbf{F}$ with $m - n$ zero columns and use Algorithm 25 to compute $\mathbf{PF}$; according to Proposition 12.5, this uses $\mathcal{O}(\mathsf{MM}'(m,\xi/m)) \subseteq \mathcal{O}(\mathsf{MM}'(n,\xi/n))$ operations.

Steps **5.a** and **5.b**: Computing $\mathbf{G}$ involves no arithmetic operation since the product $\mathbf{PF}$ has already been computed in Step **3**; $\mathbf{G}$ has row degree bounded by $\mathbf{t}$ (componentwise). Let us denote $\hat{m}$ the number of rows of $\mathbf{P}_2$. Because both $\mathbf{P}$ and $\mathbf{F}$ have full rank and $\mathbf{P}_1\mathbf{F} = \mathbf{0}$, $\mathbf{G}$ has full rank and at least n rows in $\mathbf{P}$ are not in the kernel of $\mathbf{F}$, which means $n \leqslant \hat{m}$. Furthermore, according to [ZLS12, Theorem 3.6], we have $\hat{m} \leqslant 3n/2$. Then, $\mathbf{G}$ is an $\hat{m} \times n$ matrix with $n \leqslant \hat{m} \leqslant 3n/2$ and with row degree bounded by $\mathbf{t}$. In addition, we have $\mathbf{t} \leqslant \mathbf{s}$ [ZLS12, Lemma 3.12], and thus in particular $|\mathbf{t}| \leqslant \xi$.

Algorithm 26 – MinKerBas
(Shifted minimal kernel basis [ZLS12])

Input:

- matrix $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ with full rank and $m \geqslant n$,
- a non-decreasing shift $\mathbf{s} \in \mathbb{Z}^m$ such that $\operatorname{rdeg}(\mathbf{F}) \leqslant \mathbf{s}$.

Output:

- an $\mathbf{s}$-minimal kernel basis $\mathbf{N}$ for $\mathbf{F}$,
- the $\mathbf{s}$-row degree of $\mathbf{N}$.

1. $\rho \leftarrow \sum_{i=m-n+1}^{m} s_i$ and $\lambda \leftarrow \lceil \rho/n \rceil$

2. $\mathbf{P} \leftarrow$ DaCAppBas$((3\lambda, \ldots, 3\lambda), \mathbf{F}, \mathbf{s})$
$\mathbf{P} \leftarrow$ permute the rows of $\mathbf{P}$ so that $\operatorname{rdeg}_{\mathbf{s}}(\mathbf{P})$ is non-decreasing

3. Write $\mathbf{P} = [\mathbf{P}_1^\mathsf{T} \;\; \mathbf{P}_2^\mathsf{T}]^\mathsf{T}$ where $\mathbf{P}_1$ consists of all rows $\mathbf{p}$ of $\mathbf{P}$ satisfying $\mathbf{pF} = 0$

4. If $n = 1$ then return $(\mathbf{P}_1, \operatorname{rdeg}_{\mathbf{s}}(\mathbf{P}_1))$

5. Else:

- **a.** $\mathbf{t} \leftarrow \operatorname{rdeg}_{\mathbf{s}}(\mathbf{P}_2) - (3\lambda, \ldots, 3\lambda)$
- **b.** $\mathbf{G} \leftarrow X^{-3\lambda}\mathbf{P}_2\mathbf{F}$
- **c.** Write $\mathbf{G} = [\mathbf{G}_1 \;\; \mathbf{G}_2]$ where $\mathbf{G}_1$ has $\lfloor n/2 \rfloor$ columns and $\mathbf{G}_2$ has $\lceil n/2 \rceil$ columns
- **d.** $(\mathbf{N}_1, \mathbf{u}) \leftarrow$ MinKerBas$(\mathbf{G}_1, \mathbf{t})$
- **e.** $(\mathbf{N}_2, \mathbf{v}) \leftarrow$ MinKerBas$(\mathbf{N}_1\mathbf{G}_2, \mathbf{u})$
- **f.** $\mathbf{N} \leftarrow [\mathbf{P}_1^\mathsf{T} \;\; (\mathbf{N}_2\mathbf{N}_1\mathbf{P}_2)^\mathsf{T}]^\mathsf{T}$
- **g.** Return $(\mathbf{N}, (\operatorname{rdeg}_{\mathbf{s}}(\mathbf{P}_1), \mathbf{v}))$

Step **5.c**: for the recursive calls of Steps **5.d** and **5.e**, we will need to check that our assumptions on the dimensions, the degrees, and the rank of the input are maintained. Here, we first remark that $\mathbf{G}_1$ and $\mathbf{G}_2$ have full rank and respective dimensions $\hat{m} \times \lfloor n/2 \rfloor$ and $\hat{m} \times \lceil n/2 \rceil$, with $\hat{m} \geqslant \lceil n/2 \rceil \geqslant \lfloor n/2 \rfloor$. Their row degrees are bounded by $\mathbf{t}$, which is in non-decreasing order and satisfies $|\mathbf{t}| \leqslant \xi$.

Step **5.d**: $\mathbf{N}_1$ is a $\mathbf{t}$-minimal kernel basis of $\mathbf{G}_1$ and therefore it has $\hat{m} - \lfloor n/2 \rfloor$ rows and $\hat{m}$ columns. Besides, $\mathbf{u} = \mathrm{rdeg}_{\mathbf{t}}(\mathbf{N}_1)$ and by [ZLS12, Theorem 3.4], we have $|\mathbf{u}| \leqslant |\mathbf{t}| \leqslant \xi$.

Step **5.e**: we remark that $\mathbf{N}_1\mathbf{G}_2$ has $\lceil n/2 \rceil$ columns and $\hat{m} - \lfloor n/2 \rfloor \geqslant \lceil n/2 \rceil$ rows. We now show that it has full rank. Let us consider $\hat{\mathbf{N}}_2$ any $\mathbf{u}$-minimal kernel basis for $\mathbf{N}_1\mathbf{G}_2$. Then $\hat{\mathbf{N}}_2$ has $\hat{m} - \lfloor n/2 \rfloor - r$ rows, where r is the rank of $\mathbf{N}_1\mathbf{G}_2$. Our goal is to prove that $r = \lceil n/2 \rceil$. The matrix $\hat{\mathbf{N}} = [\mathbf{P}_1^{\mathsf{T}} | (\hat{\mathbf{N}}_2\mathbf{N}_1\mathbf{P}_2)^{\mathsf{T}}]^{\mathsf{T}}$ is an $\mathbf{s}$-minimal kernel basis for $\mathbf{F}$ [ZLS12, Theorems 3.9 and 3.15]. In particular, since $\mathbf{F}$ has full rank, $\hat{\mathbf{N}}$ has $m - n$ rows. Since $\mathbf{P}_1$ has $m - \hat{m}$ rows, this gives $m - n = m - \hat{m} + \hat{m} - \lfloor n/2 \rfloor - r = m - \lfloor n/2 \rfloor - r$. Thus $n = \lfloor n/2 \rfloor + r$, and $r = \lceil n/2 \rceil$.

Furthermore, $\mathrm{rdeg}(\mathbf{G}_2) \leqslant \mathbf{t}$ and $\mathrm{rdeg}_{\mathbf{t}}(\mathbf{N}_1) = \mathbf{u}$, hence

$$\mathrm{rdeg}(\mathbf{N}_1\mathbf{G}_2) \leqslant \mathrm{rdeg}_{\mathrm{rdeg}(\mathbf{G}_2)}(\mathbf{N}_1) \leqslant \mathrm{rdeg}_{\mathbf{t}}(\mathbf{N}_1) = \mathbf{u}.$$

We have $|\mathbf{t}| \leqslant \xi$ and $|\mathbf{u}| \leqslant \xi$. Augmenting $\mathbf{N}_1$ and $\mathbf{G}_2$ with zero entries so that their dimensions are $\hat{m} \times \hat{m}$, by Proposition 12.5 the product $\mathbf{N}_1\mathbf{G}_2$ can be computed using $\mathcal{O}(\mathsf{MM}'(\hat{m}, \xi/\hat{m})) \subseteq \mathcal{O}(\mathsf{MM}'(n, \xi/n))$ operations. Then, $\mathbf{N}_2$ is a $\mathbf{t}$-minimal kernel basis for $\mathbf{N}_1\mathbf{G}_2$; it has $\hat{m} - n$ rows and $\hat{m} - \lceil n/2 \rceil$ columns, its $\mathbf{u}$-row degree is $\mathbf{v} = \mathrm{rdeg}_{\mathbf{u}}(\mathbf{N}_2)$, and we have $|\mathbf{v}| \leqslant |\mathbf{u}| \leqslant \xi$ [ZLS12, Theorem 3.4].

Step **5.f**: using the previously given dimensions and degree bounds for $\mathbf{N}_1$ and $\mathbf{N}_2$, one can easily check that the product $\mathbf{N}_2\mathbf{N}_1$ can be computed by Algorithm 25 using $\mathcal{O}(\mathsf{MM}'(\hat{m}, \xi/\hat{m})) \subseteq \mathcal{O}(\mathsf{MM}'(n, \xi/n))$ operations. Now, $\mathbf{P}_2$ is $\hat{m} \times m$ with $m \geqslant \hat{m}$, and denoting $\mathbf{w}' = \mathbf{t} + (3\lambda, \ldots, 3\lambda)$, $\mathbf{P}_2$ has its row degree bounded by $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}_2) = \mathbf{w}'$, with $|\mathbf{w}'| = |\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}_2)| \leqslant |\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})| \leqslant 7\xi$. Besides, $|\mathrm{rdeg}_{\mathbf{w}'}(\mathbf{N}_2\mathbf{N}_1)| \leqslant |\mathrm{rdeg}_{\mathbf{t}}(\mathbf{N}_2\mathbf{N}_1)| + 3(\hat{m} - n)\lambda \leqslant |\mathbf{v}| + 3n\lambda/2 \leqslant 4\xi$. Then, the product $\mathbf{N}_2\mathbf{N}_1\mathbf{P}_2$ can be computed with Algorithm 25 using $\mathcal{O}(m/\hat{m}\mathsf{MM}'(\hat{m}, \xi/\hat{m})) \subseteq \mathcal{O}(\mathsf{MM}'(n, \xi/n))$ operations, since $m \in \mathcal{O}(n)$ and $n \leqslant \hat{m}$.

Thus, we have two recursive calls with half the column dimension and the same bound ξ, and additional $\mathcal{O}(\mathsf{MM}'(n, \xi/n) + \mathsf{MM}''(n, \xi/n))$ operations for the matrix products and the computation of a minimal basis of Hermite-Padé approximants. Overall Algorithm 26 uses $\mathcal{O}(\overline{\mathsf{MM}'}(n, \xi/n) + \overline{\mathsf{MM}''}(n, \xi/n))$ operations: since $n \in \Theta(m)$, we obtain the announced cost estimate; the upper bound is a direct consequence of Lemma 12.3. ☙

13

Computing shifted Popov interpolant bases

In this chapter, we detail our fast algorithms for computing interpolant bases, which are relation bases as in Problem 4 in the case of a Jordan multiplication matrix. We have seen in Section 2.4.1 that this problem can be rewritten as a system of n modular equations in m unknowns. As such, it may be seen as a specific case of Problem 9 for moduli which are known through their roots and multiplicities, which correspond to the diagonal entries and block sizes of the Jordan matrix.

In Chapter 8, we gave an algorithm for Problem 9 with cost bound $\tilde{\mathcal{O}}(m^{\omega-1}D)$, under the assumption that $n \in \mathcal{O}(m)$ and where D is the dimension of the multiplication matrix. In our context here, we manage to remove this assumption, in particular by exploiting the knowledge of the roots and multiplicities of the moduli in the residual computation, and by introducing a change of shift technique to control the degrees in the recursion.

In this chapter, we only present the global structure of our algorithms, leaving the details of some technical procedures to the next Chapter 14.

13.1 Divide-and-conquer approach for a triangular multiplication matrix

In Section 6.4, we presented an algorithm from [BL00] which iteratively computes relation bases in shifted Popov form when the multiplication matrix is triangular. In this section, we show how this can be turned into a divide-and-conquer algorithm, which can be seen as an generalization of [BL94, Algorithm SPHPS].

Our goal here is only to present this approach and its correctness; we are not interested in its efficiency, since for an arbitrary triangular multiplication matrix it is not as efficient as our general algorithm based on linear algebra in Chapter 4. In Chapters 13 and 14, we will study the specific case of a multiplication matrix in Jordan form, for which we use the same divide-and-conquer scheme along with some additional ingredients to obtain fast algorithms.

Algorithm 27 – DACRELBAS
(Divide-and-conquer relation basis for triangular mult. mat.)

Input:

- an upper triangular matrix $\mathbf{M} \in \mathbb{K}^{D \times D}$,
- a matrix $\mathbf{F} \in \mathbb{K}^{m \times D}$,
- a shift $\mathbf{s} \in \mathbb{Z}^m$.

Output: an $\mathbf{s}$-ordered weak Popov relation basis of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$.

1. If $D = 1$ then return ITERRELBAS$(\mathbf{M}, \mathbf{F}, \mathbf{s})$

2. Else:

- **a.** $\mathbf{M}^{(1)} \leftarrow$ leading principal $\lfloor D/2 \rfloor \times \lfloor D/2 \rfloor$ submatrix of $\mathbf{M}$
- **b.** $\mathbf{F}^{(1)} \leftarrow$ first $\lfloor D/2 \rfloor$ columns of $\mathbf{F}$
- **c.** $\mathbf{P}^{(1)} \leftarrow$ DACRELBAS$(\mathbf{M}^{(1)}, \mathbf{F}^{(1)}, \mathbf{s})$
- **d.** $\mathbf{G} \leftarrow$ last $\lceil D/2 \rceil$ columns of $\mathbf{P}^{(1)} \cdot \mathbf{F}$
- **e.** $\mathbf{M}^{(2)} \leftarrow$ trailing principal $\lceil D/2 \rceil \times \lceil D/2 \rceil$ submatrix of $\mathbf{M}$
- **f.** $\mathbf{P}^{(2)} \leftarrow$ DACRELBAS$(\mathbf{M}^{(2)}, \mathbf{G}, \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}^{(1)}))$
- **g.** Return $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$

Proposition 13.1. *Assuming that $\mathbf{M} \in \mathbb{K}^{D \times D}$ is an upper triangular matrix, Algorithm 27 solves Problem 4 and returns a relation basis in $\mathbf{s}$-ordered weak Popov form.*

Proof. For the correctness of the base case of the recursion at Step **1**, we refer to Proposition 6.5. Note that the basis returned by Step **1** is in $\mathbf{s}$-Popov form, and thus in particular in $\mathbf{s}$-ordered weak Popov form.

Concerning the recursion, our proof of correctness fundamentally relies on Theorem 1.28. Using notation from that theorem, here, we consider the module of relations $\mathcal{M} = \mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$ for which we want to find a basis, and the module $\mathcal{M}^{(1)} = \mathrm{Syz}_{\mathbf{M}^{(1)}}(\mathbf{F}^{(1)})$. Since $\mathbf{M}$ is upper triangular, we have $\mathcal{M} \subseteq \mathcal{M}^{(1)}$.

Let us assume that $\mathbf{P}^{(1)}$ and $\mathbf{P}^{(2)}$, as computed by the recursive calls at Step **2.c** and **2.f**, are $\mathbf{s}$- and $\mathbf{t}$-ordered weak Popov relation basis of $\mathrm{Syz}_{\mathbf{M}^{(1)}}(\mathbf{F}^{(1)})$ and $\mathrm{Syz}_{\mathbf{M}^{(2)}}(\mathbf{G})$, respectively, where $\mathbf{t} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}^{(1)})$.

We write the input $\mathbf{F}$ as $\mathbf{F} = [\mathbf{F}^{(1)} \ \ *]$, with $\mathbf{F}^{(1)}$ consisting of the first $\lfloor D/2 \rfloor$ columns of $\mathbf{F}$ as in Step **2**. Since $\mathbf{M}$ is upper triangular, from the fact that the rows of $\mathbf{P}^{(1)}$ are relations of $\mathrm{Syz}_{\mathbf{M}^{(1)}}(\mathbf{F}^{(1)})$ we obtain that $\mathbf{P}^{(1)} \cdot \mathbf{F} = [\mathbf{0} \ \ *]$. Then, by construction of $\mathbf{G}$ at Step **2.d**, we have $\mathbf{P}^{(1)} \cdot \mathbf{F} = [\mathbf{0} \ \ \mathbf{G}]$.

As a consequence, the module $\mathcal{M}^{(2)}$ as defined in Theorem 1.28 is

$$\begin{aligned}\mathcal{M}^{(2)} &= \{\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m} \mid \boldsymbol{\lambda}\mathbf{P}^{(1)} \in \mathcal{M}\}\\ &= \{\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m} \mid \boldsymbol{\lambda}\mathbf{P}^{(1)} \cdot \mathbf{F} = \mathbf{0}\}\\ &= \{\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m} \mid \boldsymbol{\lambda} \cdot [\mathbf{0} \;\; \mathbf{G}] = \mathbf{0}\}\\ &= \mathrm{Syz}_{\mathbf{M}^{(2)}}(\mathbf{G}),\end{aligned}$$

where again the last equality follows from the fact that $\mathbf{M}$ is upper triangular. Thus, $\mathbf{P}^{(2)}$ is a basis of $\mathcal{M}^{(2)}$.

To summarize, $\mathbf{P}^{(1)}$ is a $\mathbf{s}$-ordered weak Popov basis of $\mathcal{M}^{(1)}$, and $\mathbf{P}^{(2)}$ is a $\mathbf{t}$-ordered weak Popov basis of $\mathcal{M}^{(2)}$. Then, from the items (i) and (iii) of Theorem 1.28, it follows that $\mathbf{P}^{(2)}\mathbf{P}^{(1)}$ is a $\mathbf{0}$-ordered weak Popov basis of $\mathcal{M}$. Hence the correctness. ☙

13.2 Fast interpolant bases in reduced form for almost uniform shifts

In this section, we provide a description of our divide-and-conquer algorithm to compute minimal interpolant for the case of a shift whose entries are close to uniform. As summarized in Section 2.4.2, we obtain an algorithm in $\tilde{\mathcal{O}}(m^{\omega-1}D)$ by using several new ingredients.

Instead of resorting to kernel bases as for Problem 9, we directly follow the divide-and-conquer approach presented above in Algorithm 27. For efficiency, we do not stop the recursion at $D = 1$ but rather at $D \approx m$, in which case we solve the problem via the algorithm of Chapter 4 based on fast dense linear algebra. Furthermore, to ensure that shifts stay almost uniform in recursive calls, we resort to changes of shifts at each node of the recursion; and we rely on a fast algorithm for computing residuals in this context, using techniques involving Chinese remainder evaluation and interpolation.

As a parenthesis, regarding the introduction of this chapter, we recall that in the case of systems of linear modular equations, the efficient use of kernel bases and the fast computation of residuals were two components that pushed us to require that the number of equations be not much larger than the number of unknowns (see Sections 6.3 and 8.1).

In this section, we assume that $\mathbf{J} \in \mathbb{K}^{D\times D}$ is a Jordan matrix given by means of a standard representation as in Eq. (2.3). Our algorithm relies on two subroutines for which cost estimates are given in Chapter 14 and are taken for granted here:

- in Section 14.1, the *change of shift*: given an $\mathbf{s}$-minimal interpolant basis $\mathbf{P}$ and some shift $\mathbf{t}$, compute an $(\mathbf{s}+\mathbf{t})$-minimal interpolant basis;
- in Section 14.2, the fast computation of a *residual*, that is, a product of the form $\mathbf{P} \cdot \mathbf{F}$ defined by the multiplication matrix as in Section 2.1.2.

It also uses algorithms that have been given previously in this document:

- in Chapter 4, the computation of an $\mathbf{s}$-minimal interpolant basis using linear algebra, which is used here for the base case of the recursion;

- in Section 12.2, the fast multiplication of two polynomial matrices with respect to the average row degree of the operands and of the result.

Algorithm 28 – MinIntBas
(Minimal interpolant basis)

Input:
- a Jordan matrix $\mathbf{J} \in \mathbb{K}^{D \times D}$ in standard representation,
- a matrix $\mathbf{F} \in \mathbb{K}^{m \times D}$.

Output: a $\mathbf{0}$-minimal interpolant basis of $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$.

1. If $D \leqslant m$ then return LinPopovRelBas$(\mathbf{J}, \mathbf{F}, \mathbf{0})$

2. Else:

a. $\mathbf{J}^{(1)} \leftarrow$ leading principal $\lfloor D/2 \rfloor \times \lfloor D/2 \rfloor$ submatrix of $\mathbf{J}$

b. $\mathbf{F}^{(1)} \leftarrow$ first $\lfloor D/2 \rfloor$ columns of $\mathbf{F}$

c. $\mathbf{P}^{(1)} \leftarrow$ MinIntBas$(\mathbf{J}^{(1)}, \mathbf{F}^{(1)})$

d. $\mathbf{G} \leftarrow$ last $\lceil D/2 \rceil$ columns of $\mathbf{P}^{(1)} \cdot \mathbf{F} =$ JordanMul$(\mathbf{J}, \mathbf{P}^{(1)}, \mathbf{F})$

e. $\mathbf{J}^{(2)} \leftarrow$ trailing principal $\lceil D/2 \rceil \times \lceil D/2 \rceil$ submatrix of $\mathbf{J}$

f. $\mathbf{P}^{(2)} \leftarrow$ MinIntBas$(\mathbf{J}^{(2)}, \mathbf{G})$

g. $\mathbf{R} \leftarrow$ ChangeShift$(\mathbf{P}^{(2)}, \mathbf{0}, \mathrm{rdeg}(\mathbf{P}^{(1)}))$

h. Return RDegPolMatMul$(\mathbf{P}^{(1)}, \mathbf{R}, D)$

Using notation in Algorithm 28, we remark that the leading and trailing principal submatrices $\mathbf{J}^{(1)}$ and $\mathbf{J}^{(2)}$ are still in Jordan canonical form, albeit not necessarily in standard representation; this can however be restored by a single pass through the array.

Proposition 13.2. *Algorithm 28 is correct, and for $D > m$ it uses*

$$\begin{aligned}&\mathcal{O}(m^{\omega-1}\mathsf{M}(D) + m^{\omega}\mathsf{M}(D/m)\log(D/m)^2 + m^{\omega-1}D\log(m) + m\mathsf{M}(D)\log(D)\log(D/m)) \\ &\quad \subseteq \mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(D)\log(D/m))\end{aligned}$$

operations in $\mathbb{K}$. If $D \leqslant m$, it uses $\mathcal{O}(mD^{\omega-1} + D^{\omega}\log(D))$ operations in $\mathbb{K}$.

Proof. Concerning the base case $D \leqslant m$, the correctness and the cost bound both follow from Proposition 4.18 and the corresponding Algorithm 3.

Let us consider the case of $D > m$. For the correctness of the recursive procedure, our proof relies on Theorem 1.28 and is also close to the proof of Proposition 13.1. We consider the modules $\mathcal{M} = \mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$ and $\mathcal{M}^{(1)} = \mathrm{Syz}_{\mathbf{J}^{(1)}}(\mathbf{F}^{(1)})$. We assume that $\mathbf{P}^{(1)}$, as computed by the first recursive call, is a $\mathbf{0}$-minimal interpolant basis of $\mathrm{Syz}_{\mathbf{J}^{(1)}}(\mathbf{F}^{(1)})$.

Writing the input $\mathbf{F}$ as $\mathbf{F} = [\mathbf{F}^{(1)} \ \ *]$, since $\mathbf{J}$ is upper triangular, the fact that the rows $\mathbf{P}^{(1)}$ are interpolants of $\mathrm{Syz}_{\mathbf{J}^{(1)}}(\mathbf{F}^{(1)})$ implies that $\mathbf{P}^{(1)} \cdot \mathbf{F} = [\mathbf{0} \ \ *]$. Then, by construction of $\mathbf{G}$ at Step **2.d**, we have $\mathbf{P}^{(1)} \cdot \mathbf{F} = [\mathbf{0} \ \ \mathbf{G}]$.

As a consequence, the module $\mathcal{M}^{(2)}$ as defined in Theorem 1.28 is

$$\begin{aligned}\mathcal{M}^{(2)} &= \{\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m} \mid \boldsymbol{\lambda}\mathbf{P}^{(1)} \in \mathrm{Syz}_{\mathbf{J}}(\mathbf{F})\}\\ &= \{\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m} \mid \boldsymbol{\lambda}\mathbf{P}^{(1)} \cdot \mathbf{F} = \mathbf{0}\}\\ &= \{\boldsymbol{\lambda} \in \mathbb{K}[X]^{1\times m} \mid \boldsymbol{\lambda} \cdot [\mathbf{0} \;\; \mathbf{G}] = \mathbf{0}\}\\ &= \mathrm{Syz}_{\mathbf{J}^{(2)}}(\mathbf{G}),\end{aligned}$$

where again the last equality follows from the fact that $\mathbf{J}$ is upper triangular.

Let us then denote by $\mathbf{P}^{(2)}$ the matrix computed by the second recursive call, which is a $\mathbf{0}$-minimal interpolant basis of $\mathrm{Syz}_{\mathbf{J}^{(2)}}(\mathbf{G})$ by construction. From the identity above, $\mathbf{P}^{(2)}$ is a $\mathbf{0}$-reduced basis of $\mathcal{M}^{(2)}$. Moreover, Proposition 14.2 indicates that $\mathbf{R}$ is a $\mathbf{t}$-reduced form of $\mathbf{P}^{(2)}$, where $\mathbf{t} = \mathrm{rdeg}(\mathbf{P}^{(1)})$.

To summarize, $\mathbf{P}^{(1)}$ is a $\mathbf{0}$-reduced basis of $\mathcal{M}^{(1)}$, and $\mathbf{R}$ is a $\mathbf{t}$-reduced basis of $\mathcal{M}^{(2)}$. Then, from Theorem 1.28, it follows that $\mathbf{R}\mathbf{P}^{(1)}$ is a $\mathbf{0}$-reduced basis of $\mathcal{M}$, or in other words, a $\mathbf{0}$-minimal interpolant basis of $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$. The correctness follows, since $\mathbf{R}\mathbf{P}^{(1)}$ is the product computed at Step **2.h**.

Let us now prove the announced cost bound for $D > m$. Without loss of generality, we assume that D/m is a power of 2. Each of Steps **2.c** and **2.f** calls the algorithm recursively on an instance of Problem 4 with dimensions m and $D/2$.

Let us first give some remarks on the degrees in the manipulated matrices. From Lemma 2.10, we obtain

$$\deg(\det(\mathbf{P})) \leqslant D, \quad \deg(\det(\mathbf{P}^{(1)})) \leqslant D/2, \quad \text{and } \deg(\det(\mathbf{P}^{(2)})) \leqslant D/2.$$

According to the item (v) of Theorem 1.11, the sum of the row degrees of a $\mathbf{0}$-reduced matrix equals the degree of its determinant, hence

$$|\mathrm{rdeg}(\mathbf{P})| \leqslant D, \quad |\mathrm{rdeg}(\mathbf{P}^{(2)})| \leqslant D/2, \quad \text{and } |\mathbf{d}| = |\mathrm{rdeg}(\mathbf{P}^{(1)})| \leqslant D/2.$$

Then, we analyze the contribution of each step to the total cost.

- The leaves of the recursion are for the dimensions $m/2 \leqslant D \leqslant m$, and therefore by Proposition 4.18 each of them uses $\mathcal{O}(m^{\omega}\log(m))$ operations. For $D > m$, with D/m a power of 2, the recursion leads to D/m leaves, which thus yield a total cost of $\mathcal{O}(m^{\omega-1}D\log(m))$ operations.

- According to Proposition 14.3, Step **2.d** uses

  $$\mathcal{O}(\mathsf{MM}(m, D/m)\log(D/m) + m\mathsf{M}(D)\log(D))$$

  operations. Using the super-linearity property $\mathcal{H}_{\mathsf{MM}(\cdot,\cdot)}$ of Eq. (6.2), this contributes to the total cost as $\mathcal{O}(\mathsf{MM}(m, D/m)\log(D/m)^2 + m\mathsf{M}(D)\log(D)\log(D/m))$ operations.

- For Step **2.g**, we use Proposition 14.2 with $\xi = D$, remarking that both the sum of the entries of $\mathbf{t} = \mathrm{rdeg}(\mathbf{P}^{(1)})$ and that of $\mathrm{rdeg}(\mathbf{P}^{(2)})$ are at most $D/2$. Then, the

change of shift is performed using $\mathcal{O}(\overline{\mathsf{MM}'}(m, D/m) + \overline{\mathsf{MM}''}(m, D/m))$ operations. Thus, altogether the time spent in this step is

$$\mathcal{O}\left(\sum_{0 \leqslant i \leqslant \log(D/m)} 2^i \left(\overline{\mathsf{MM}'}(m, 2^{-i}D/m) + \overline{\mathsf{MM}''}(m, 2^{-i}D/m)\right)\right)$$

operations; we give an upper bound for this quantity in Lemma 12.4.

- From Proposition 14.2 we obtain that $\mathrm{rdeg}_{\mathbf{t}}(\mathbf{R}) \leqslant |\mathrm{rdeg}(\mathbf{P}^{(2)})| + |\mathbf{t}| \leqslant D$. Then, using Proposition 12.5 with $\xi = D$, the polynomial matrix multiplication in Step **2.f** can be done in time $\mathcal{O}(\mathsf{MM}'(m, D/m))$. Besides, it is easily verified that by definition $\mathsf{MM}'(m, D/m) \leqslant \overline{\mathsf{MM}'}(m, D/m)$, so that the cost for this step is dominated by the cost for the change of shift.

Adding these costs and using the bounds in Section 6.1 leads to the conclusion. ☙

Now, if the shift in input is not uniform, we can always resort to this divide-and-conquer approach with the uniform shift and eventually perform a change of shift to obtain the sought *shifted* minimal interpolant basis.

Algorithm 29 – ShiftMinIntBas
(Shifted minimal interpolant basis)

Input:
- a Jordan matrix $\mathbf{J} \in \mathbb{K}^{D \times D}$ in standard representation,
- a matrix $\mathbf{F} \in \mathbb{K}^{m \times D}$,
- a shift $\mathbf{s} \in \mathbb{Z}^m$.

Output: an $\mathbf{s}$-minimal interpolant basis of $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$.

1. If $D \leqslant m$ then return LinPopovRelBas$(\mathbf{J}, \mathbf{F}, \mathbf{s})$

2. Else:

  a. $\mathbf{P} \leftarrow$ MinIntBas$(\mathbf{J}, \mathbf{F})$

  b. Return ChangeShift$(\mathbf{P}, \mathbf{0}, \mathbf{s} - \min(\mathbf{s}))$

We now prove the following cost bound, which refines that in Theorem 2.19.

Proposition 13.3. *Let $\xi = |\mathbf{s} - \min(\mathbf{s})|$. Algorithm 29 is correct, and for $D > m$ it uses*

$$\begin{aligned}&\mathcal{O}(m^{\omega-1}\mathsf{M}(D) + m^{\omega}\mathsf{M}(D/m)\log(D/m)^2 + m^{\omega-1}D\log(m) + m\mathsf{M}(D)\log(D)\log(D/m)\\ &\quad + m^{\omega-1}\mathsf{M}(\xi) + m^{\omega}\mathsf{M}(\xi/m)\log(\xi/m))\\ &\subseteq \mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(D)\log(D/m) + m^{\omega-1}\mathsf{M}(\xi) + m^{\omega}\mathsf{M}(\xi/m)\log(\xi/m))\end{aligned}$$

operations in $\mathbb{K}$. If $D \leqslant m$, it uses $\mathcal{O}(mD^{\omega-1} + D^{\omega}\log(D))$ operations in $\mathbb{K}$.

Proof. The correctness of Algorithm 29 follows from the correctness of Algorithms 3, 28 and 31. We remark that at Step **2.b** we use the shift $\mathbf{s} - \min(\mathbf{s})$ instead of $\mathbf{s}$, in order to ensure the requirement of ChangeShift that the input shift should have nonnegative entries.

Concerning the cost bound when $D > m$, Proposition 13.2 gives the number of operations used by Step **2.a** to produce $\mathbf{P}$, which satisfies $|\mathrm{rdeg}(\mathbf{P})| = \deg(\det(\mathbf{P})) \leqslant D$ according to Lemma 2.10. As a consequence, we have $|\mathrm{rdeg}(\mathbf{P})| + |\mathbf{s} - \min(\mathbf{s})| \leqslant D + \xi$: Proposition 14.2 states that Step **2.b** can be performed using

$$\mathcal{O}(\overline{\mathsf{MM}'}(m, (D+\xi)/m) + \overline{\mathsf{MM}''}(m, (D+\xi)/m))$$

field operations. The cost bound then follows from the bounds in Lemma 12.3. ∎

13.3 Fast interpolant bases in Popov form for arbitrary shifts

In the previous section, we have given an algorithm in $\mathcal{O}\tilde{}(m^{\omega-1}D)$ for almost uniform shifts. Still, the situation is not yet completely satisfactory: for worst case shifts, the cost bound for this algorithm is far from our target cost.

For example, for a shift $\mathbf{s}$ that has a large amplitude $\max(\mathbf{s}) - \min(\mathbf{s})$, such as the Hermite shift that we discussed in Section 1.2.2, the cost bound may become as large as $\mathcal{O}\tilde{}(m^{\omega+1}D)$. The main reason behind this high cost is that performing a change of shift is costly for such shifts. As a comparison, even the simple divide-and-conquer approach of Section 13.1 is more efficient in this case: assuming that the residual matrix $\mathbf{G}$ is computed efficiently (using for example the algorithm of Section 14.2), Algorithm 27 uses $\mathcal{O}\tilde{}(m^{\omega}D)$ operations for an arbitrary shift.

Here, we design an algorithm which computes the interpolant basis in shifted Popov form using $\mathcal{O}\tilde{}(m^{\omega-1}D)$ for an arbitrary shift. To achieve this, we use our strategy revolving around the knowledge of the minimal degree, which was summarized in general terms in Section 1.2.1 and used in our fast algorithms for approximant bases and solution bases. More explicitly, we follow a divide-and-conquer scheme similar to that of Algorithm 27 to find the shifted minimal degree of $\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$, using this additional knowledge to consider another shift with good properties that allow us to solve the problem efficiently by relying on the fast algorithm of the previous section.

More precisely, our algorithm relies on two subroutines presented in Chapter 14, the second one being the key new ingredient for dealing with arbitrary shifts:

- JordanMul, detailed in Section 14.2, with an additional pre-processing given in Corollary 14.4 and at the end of Section 14.3, which computes the residual $\mathbf{P}^{(1)} \cdot \mathbf{F}$ from a basis $\mathbf{P}^{(1)}$ obtained recursively;
- MinDegIntBas, detailed in Section 14.3, which computes the $\mathbf{s}$-Popov interpolant basis when the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$ is known.

We temporarily take for granted the results in Propositions 14.3 and 14.11 concerning these subroutines.

Algorithm 30 – FastPopovIntBas
(Shifted Popov interpolant basis)

Input:

- a Jordan matrix $\mathbf{J} \in \mathbb{K}^{D \times D}$ in standard representation,
- a matrix $\mathbf{F} \in \mathbb{K}^{m \times D}$,
- a shift $\mathbf{s} \in \mathbb{Z}^m$.

Output: the $\mathbf{s}$-Popov interpolant basis of $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$.

1. If $D \leqslant m$ then return LinPopovRelBas$(\mathbf{J}, \mathbf{F}, \mathbf{s})$

2. Else:

a. $\mathbf{J}^{(1)} \leftarrow$ leading principal $\lfloor D/2 \rfloor \times \lfloor D/2 \rfloor$ submatrix of $\mathbf{J}$

b. $\mathbf{F}^{(1)} \leftarrow$ first $\lfloor D/2 \rfloor$ columns of $\mathbf{F}$

c. $\mathbf{P}^{(1)} \leftarrow$ FastPopovIntBas$(\mathbf{J}^{(1)}, \mathbf{F}^{(1)}, \mathbf{s})$

d. $\boldsymbol{\delta}^{(1)} \leftarrow$ diagonal degrees of $\mathbf{P}^{(1)}$

e. $\mathbf{G} \leftarrow$ last $\lceil D/2 \rceil$ columns of $\mathbf{P}^{(1)} \cdot \mathbf{F}$ `// using` JordanMul

f. $\mathbf{J}^{(2)} \leftarrow$ trailing principal $\lceil D/2 \rceil \times \lceil D/2 \rceil$ submatrix of $\mathbf{J}$

g. $\mathbf{P}^{(2)} \leftarrow$ FastPopovIntBas$(\mathbf{J}^{(2)}, \mathbf{G}, \mathbf{s} + \boldsymbol{\delta}^{(1)})$

h. $\boldsymbol{\delta}^{(2)} \leftarrow$ diagonal degrees of $\mathbf{P}^{(2)}$

i. Return MinDegIntBas$(\mathbf{J}, \mathbf{F}, \mathbf{s}, \boldsymbol{\delta}^{(1)} + \boldsymbol{\delta}^{(2)})$

We obtain the following result; note that the cost bound is essentially that of Proposition 13.2 multiplied by the depth $\mathcal{O}(\log(D/m))$.

Proposition 13.4. *Algorithm 30 is correct, and if $D > m$ it uses*

$$\begin{aligned}\mathcal{O}(&m^{\omega-1}\mathsf{M}(D)\log(D/m) + m^{\omega}\mathsf{M}(D/m)\log(D/m)^3 \\ &\qquad + m^{\omega-1}D\log(m)\log(D/m) + m\mathsf{M}(D)\log(D)\log(D/m)^2) \\ \subseteq\ &\mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(D)\log(D/m)^2)\end{aligned}$$

operations in $\mathbb{K}$. If $D \leqslant m$, it uses $\mathcal{O}(mD^{\omega-1} + D^{\omega}\log(D))$ operations in $\mathbb{K}$.

Proof. For the case $D \leqslant m$, the correctness and the cost bound of the base case at Step **1** both follow from Proposition 4.18: it uses $\mathcal{O}(mD^{\omega-1}+D^{\omega}\log(D)) \subseteq \mathcal{O}(m^{\omega}\log(m))$ operations.

Now, we consider the case $D > m$. Using the notation in the algorithm, assume that $\mathbf{P}^{(1)}$ is the $\mathbf{s}$-Popov interpolant basis for $\mathrm{Syz}_{\mathbf{J}^{(1)}}(\mathbf{F}^{(1)})$, and $\mathbf{P}^{(2)}$ is the $\mathbf{t}$-Popov interpolant basis of $\mathrm{Syz}_{\mathbf{J}^{(2)}}(\mathbf{F}^{(2)})$, where $\mathbf{t} = \mathbf{s} + \boldsymbol{\delta}^{(1)} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}^{(1)})$, and $\boldsymbol{\delta}^{(1)}$ and $\boldsymbol{\delta}^{(2)}$ are the $\mathbf{s}$- and $\mathbf{t}$-minimal degrees of $\mathrm{Syz}_{\mathbf{J}^{(1)}}(\mathbf{F}^{(1)})$ and $\mathrm{Syz}_{\mathbf{J}^{(2)}}(\mathbf{F}^{(2)})$, respectively.

Then, we can rely on Theorem 1.28 exactly like in the proof of Proposition 13.1. The item (iv) of this theorem states that the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$ is the sum $\boldsymbol{\delta}^{(1)} + \boldsymbol{\delta}^{(2)}$. As a result, Proposition 14.11 states that Step **2.i** correctly computes and returns the $\mathbf{s}$-Popov interpolant basis of $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$.

Concerning the cost bound, the recursion stops when $D \leqslant m$, and thus the algorithm uses $\mathcal{O}(m^{\omega}\log(m))$ operations. The depth of the recursion is $\mathcal{O}(\log(D/m))$; we have two recursive calls in dimensions $m \times D/2$, and two calls to subroutines with cost bounds given in Proposition 14.3 and Corollary 14.4 concerning the computation of the residual at Step **2.e**, and Proposition 14.11 concerning Step **2.i**. The conclusion follows from the super-linearity property $\mathcal{H}_{\mathsf{M}(\cdot)}$ in Eq. (6.1) ∎

14

Details of new ingredients for interpolant bases

In this chapter, we detail the three main technical tools that we introduced in order to design our fast interpolant basis algorithms described in Chapter 13. For the first algorithm for almost uniform shift, we relied on a *change of shift* at each node of the recursive tree so as to control the degrees in the bases computed recursively. For both algorithms, an essential tool is the computation of *residuals*, that is, products of the form $\mathbf{P} \cdot \mathbf{F}$ for $\mathbf{P} \in \mathbb{K}[X]^{m \times m}$ and $\mathbf{F} \in \mathbb{K}^{m \times D}$, as defined in Section 2.1. Finally, for dealing with arbitrary shifts, we use partial linearization techniques to efficiently compute the interpolant basis when the minimal degree is known.

14.1 Fast shifted reduction of a reduced matrix

In Algorithm 29, a key ingredient to achieve efficiency is to control the size of the intermediate interpolant bases that are computed in recursive calls. For this, we compute all minimal bases for the *uniform* shift and then recover the *shifted* minimal basis using what we call a change of shift, that we detail in this section. More precisely, we are interested in the problem of transforming an $\mathbf{s}$-reduced matrix $\mathbf{P} \in \mathbb{K}[X]^{m \times m}$ with full rank into a unimodularly equivalent matrix that is $(\mathbf{s}+\mathbf{t})$-reduced for some given shift $\mathbf{t} \in \mathbb{Z}^m_{\geqslant 0}$.

Compared to a general row reduction algorithm such as the one in [GSSV12], our algorithm achieves efficient computation with regards to the average row degree of the input $\mathbf{P}$ rather than the maximum degree of the entries of $\mathbf{P}$. The main consequence of having an $\mathbf{s}$-*reduced* input $\mathbf{P}$ is that no high-degree cancellation can occur when performing unimodular transformations on the rows of $\mathbf{P}$, which is formalized as the predictable-degree property (see Theorem 1.11). In particular, the unimodular transformation between $\mathbf{P}$ and an $(\mathbf{s}+\mathbf{t})$-reduced equivalent matrix has small row degree, and the proposition below shows how to exploit this to solve our problem via the computation of a shifted minimal kernel basis for some $2m \times m$ polynomial matrix (for a definition of kernel bases, see Definition 8.1).

Minimal kernel bases have often been used in the computation of reduced or normal forms of polynomial matrices, such as in [BvdHP88, BLV99, GJV03, BLV06, GS11].

Lemma 14.1. *Let* $\mathbf{s} \in \mathbb{Z}^m_{\geqslant 0}$ *and* $\mathbf{t} \in \mathbb{Z}^m_{\geqslant 0}$*, let* $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ *be* $\mathbf{s}$*-reduced and nonsingular, and define* $\mathbf{d} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})$*. Then* $\mathbf{R} \in \mathbb{K}[X]^{m\times m}$ *is an* $(\mathbf{s}+\mathbf{t})$*-reduced form of* $\mathbf{P}$ *with unimodular transformation* $\mathbf{U} = \mathbf{R}\mathbf{P}^{-1} \in \mathbb{K}[X]^{m\times m}$ *if and only if* $[\mathbf{U} \;\; \mathbf{R}\mathbf{X}^{\mathbf{s}}]$ *is a* $(\mathbf{d},\mathbf{t})$*-minimal kernel basis of* $[\mathbf{X}^{\mathbf{s}}\mathbf{P}^{\mathsf{T}} \;\; -\mathbf{I}_m]^{\mathsf{T}}$*.*

Proof. We first assume that the result holds for the uniform shift $\mathbf{s} = \mathbf{0} \in \mathbb{Z}^m_{\geqslant 0}$, and we show that the general case $\mathbf{s} \in \mathbb{Z}^m_{\geqslant 0}$ follows. Indeed, considering the $\mathbf{0}$-reduced matrix $\mathbf{P}\mathbf{X}^{\mathbf{s}}$ we have $\mathbf{d} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}) = \mathrm{rdeg}(\mathbf{P}\mathbf{X}^{\mathbf{s}})$. Hence $[\mathbf{U} \;\; \mathbf{R}]$ is a $(\mathbf{d},\mathbf{t})$-minimal kernel basis of $[\mathbf{X}^{\mathbf{s}}\mathbf{P}^{\mathsf{T}} \;\; -\mathbf{I}_m]^{\mathsf{T}}$ if and only if $\mathbf{R}$ is a $\mathbf{t}$-reduced form of $\mathbf{P}\mathbf{X}^{\mathbf{s}}$ with unimodular transformation $\mathbf{U}$ such that $\mathbf{U}\mathbf{P}\mathbf{X}^{\mathbf{s}} = \mathbf{R}$; that is, if and only if $\mathbf{R}\mathbf{X}^{-\mathbf{s}} \in \mathbb{K}[X]^{m\times m}$ is a $(\mathbf{s}+\mathbf{t})$-reduced form of $\mathbf{P}$ with unimodular transformation $\mathbf{U}$ such that $\mathbf{U}\mathbf{P} = \mathbf{R}\mathbf{X}^{-\mathbf{s}}$.

Let us now prove the proposition for the uniform shift $\mathbf{s} = \mathbf{0}$. First, we assume that $\mathbf{R} \in \mathbb{K}[X]^{m\times m}$ is a $\mathbf{t}$-reduced form of $\mathbf{P}$ with unimodular transformation $\mathbf{U}$. From $\mathbf{U}\mathbf{P} = \mathbf{R}$ it follows that the rows of $[\mathbf{U} \;\; \mathbf{R}]$ are in the kernel of $[\mathbf{P}^{\mathsf{T}} \;\; -\mathbf{I}_m]^{\mathsf{T}}$. Writing $[\mathbf{N} \;\; *]$ with $\mathbf{N} \in \mathbb{K}[X]^{m\times m}$ to denote an arbitrary basis of that kernel, we have $[\mathbf{U} \;\; \mathbf{R}] = \mathbf{V}[\mathbf{N} \;\; *]$ for some $\mathbf{V} \in \mathbb{K}[X]^{m\times m}$ and thus $\mathbf{U} = \mathbf{V}\mathbf{N}$. Since $\mathbf{U}$ is unimodular, $\mathbf{V}$ is unimodular too and $[\mathbf{U} \;\; \mathbf{R}]$ is a basis of the kernel of $[\mathbf{P}^{\mathsf{T}} \;\; -\mathbf{I}_m]^{\mathsf{T}}$. It remains to check that $[\mathbf{U} \;\; \mathbf{R}]$ is $(\mathbf{d},\mathbf{t})$-reduced. Since $\mathbf{P}$ is reduced, we have $\mathrm{rdeg}_{\mathbf{d}}(\mathbf{U}) = \mathrm{rdeg}(\mathbf{U}\mathbf{P}) = \mathrm{rdeg}(\mathbf{R})$ by the predictable-degree property in Theorem 1.11 and, using $\mathbf{t} \geqslant \mathbf{0}$, we obtain $\mathrm{rdeg}_{\mathbf{d}}(\mathbf{U}) \leqslant \mathrm{rdeg}_{\mathbf{t}}(\mathbf{R})$. Hence $\mathrm{rdeg}_{(\mathbf{d},\mathbf{t})}([\mathbf{U} \;\; \mathbf{R}]) = \mathrm{rdeg}_{\mathbf{t}}(\mathbf{R})$ and, since $\mathbf{R}$ is $\mathbf{t}$-reduced, this implies that $[\mathbf{U} \;\; \mathbf{R}]$ is $(\mathbf{d},\mathbf{t})$-reduced.

Now, let $[\mathbf{U} \;\; \mathbf{R}]$ be a $(\mathbf{d},\mathbf{t})$-minimal kernel basis for $[\mathbf{P}^{\mathsf{T}} \;\; -\mathbf{I}_m]^{\mathsf{T}}$. First, we note that $\mathbf{U}$ satisfies $\mathbf{U} = \mathbf{R}\mathbf{P}^{-1}$. It remains to check that $\mathbf{U}$ is unimodular and that $\mathbf{R}$ is $\mathbf{t}$-reduced. To do this, let $\hat{\mathbf{R}}$ denote an arbitrary $\mathbf{t}$-reduced form of $\mathbf{P}$ and let $\hat{\mathbf{U}} = \hat{\mathbf{R}}\mathbf{P}^{-1}$ be the associated unimodular transformation. From the previous paragraph, we know that $[\hat{\mathbf{U}} \;\; \hat{\mathbf{R}}]$ is a basis of the kernel of $[\mathbf{P}^{\mathsf{T}} \;\; -\mathbf{I}_m]^{\mathsf{T}}$, and since by definition $[\mathbf{U} \;\; \mathbf{R}]$ is also such a basis, we have $[\mathbf{U} \;\; \mathbf{R}] = \mathbf{W}[\hat{\mathbf{U}} \;\; \hat{\mathbf{R}}]$ for some unimodular matrix $\mathbf{W} \in \mathbb{K}[X]^{m\times m}$. In particular, $\mathbf{U} = \mathbf{W}\hat{\mathbf{U}}$ is unimodular. Furthermore, the two unimodularly equivalent matrices $[\mathbf{U} \;\; \mathbf{R}]$ and $[\hat{\mathbf{U}} \;\; \hat{\mathbf{R}}]$ are $(\mathbf{d},\mathbf{t})$-reduced, so that by definition they share the same shifted row degree up to permutation. Now, from the previous paragraph, we know that $\mathrm{rdeg}_{(\mathbf{d},\mathbf{t})}([\hat{\mathbf{U}} \;\; \hat{\mathbf{R}}]) = \mathrm{rdeg}_{\mathbf{t}}(\hat{\mathbf{R}})$, and similarly, having $\mathbf{P}$ reduced, $\mathbf{U}\mathbf{P} = \mathbf{R}$, and $\mathbf{t} \geqslant \mathbf{0}$ imply that $\mathrm{rdeg}_{(\mathbf{d},\mathbf{t})}([\mathbf{U} \;\; \mathbf{R}]) = \mathrm{rdeg}_{\mathbf{t}}(\mathbf{R})$. Thus $\mathrm{rdeg}_{\mathbf{t}}(\mathbf{R})$ and $\mathrm{rdeg}_{\mathbf{t}}(\hat{\mathbf{R}})$ are equal up to permutation, and combining this with the fact that $\mathbf{R} = \mathbf{W}\hat{\mathbf{R}}$ where $\hat{\mathbf{R}}$ is $\mathbf{t}$-reduced and $\mathbf{W}$ is unimodular, we conclude that $\mathbf{R}$ is $\mathbf{t}$-reduced. ∎

This leads to Algorithm 31; for efficient computation, it relies on the minimal kernel basis algorithm of [ZLS12], which we recalled as Algorithm 26.

Proposition 14.2. *Let* $\mathbf{s} \in \mathbb{Z}^m_{\geqslant 0}$ *and* $\mathbf{t} \in \mathbb{Z}^m_{\geqslant 0}$*, let* $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ *have full rank and be* $\mathbf{s}$*-reduced, and define* $\mathbf{d} = \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})$*. We write* ξ *to denote a parameter such that* $\xi \geqslant m$ *and* $|\mathbf{d}|+|\mathbf{t}| \leqslant \xi$*. Then, an* $(\mathbf{s}+\mathbf{t})$*-reduced form* $\mathbf{R} \in \mathbb{K}[X]^{m\times m}$ *of* $\mathbf{P}$ *and the corresponding unimodular transformation* $\mathbf{U} = \mathbf{R}\mathbf{P}^{-1} \in \mathbb{K}[X]^{m\times m}$ *can be computed using*

$$\begin{aligned}&\mathcal{O}(\overline{\mathsf{MM}'}(m,\xi/m) + \overline{\mathsf{MM}''}(m,\xi/m))\\ &\quad\subseteq\ \mathcal{O}(m^{\omega-1}\mathsf{M}(\xi) + m^{\omega}\mathsf{M}(\xi/m)\log(\xi/m))\end{aligned}$$

operations in $\mathbb{K}$*. Besides, we have* $|\mathrm{rdeg}_{\mathbf{s}+\mathbf{t}}(\mathbf{R})| = |\mathbf{d}| + |\mathbf{t}|$*.*

Algorithm 31 – ChangeShift
(Shifted reduced form of a reduced matrix)

Input:

- a nonsingular matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$,
- a nonnegative shift $\mathbf{s} \in \mathbb{Z}_{\geqslant 0}^m$ such that $\mathbf{P}$ is $\mathbf{s}$-reduced,
- a nonnegative shift $\mathbf{t} \in \mathbb{Z}_{\geqslant 0}^m$.

Output: an $(\mathbf{s}+\mathbf{t})$-reduced form of $\mathbf{P}$.

1. $\mathbf{d} \leftarrow \mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})$
2. $[\mathbf{U}\ \ \mathbf{R}] \leftarrow \textsc{MinKerBas}\left(\begin{bmatrix}\mathbf{P}\mathbf{X}^{\mathbf{s}}\\ -\mathbf{I}_m\end{bmatrix}, (\mathbf{d},\mathbf{t})\right)$
3. Return $\mathbf{R}\mathbf{X}^{-\mathbf{s}}$

Proof. Write $\mathbf{u} = (\mathbf{d},\mathbf{t})$ and $\mathbf{Q} = [\mathbf{X}^{\mathbf{s}}\mathbf{P}^{\mathsf{T}}\ \ -\mathbf{I}_m]^{\mathsf{T}}$. According to Lemma 14.1, Algorithm 31 is correct: it computes $[\mathbf{U}\ \ \mathbf{R}]$ a $\mathbf{u}$-minimal kernel basis for $\mathbf{Q}$, and returns $\mathbf{R}\mathbf{X}^{-\mathbf{s}}$ which is an $(\mathbf{s}+\mathbf{t})$-reduced form of $\mathbf{P}$. For a fast solution, the minimal kernel basis can be computed using [ZLS12, Algorithm 1], which we have rewritten in Section 12.3 (Algorithm 26) along with a detailed cost analysis.

Here, we show that the requirements of this algorithm on its input are fulfilled in our context. Concerning the input matrix, we note that $\mathbf{Q}$ has more rows than columns, and $\mathbf{Q}$ has full rank since by assumption $\mathbf{P}$ has full rank. Now, considering the requirement on the input shift, first, each element of the shift $\mathbf{u}$ bounds the corresponding row degree of $\mathbf{Q}$; and second, the rows of $\mathbf{Q}$ can be permuted before the kernel computation so as to have $\mathbf{u}$ non-decreasing, and then the columns of the obtained kernel basis can be permuted back to the original order. In details, we first compute $\mathbf{v}$ being the tuple $\mathbf{u}$ sorted in non-decreasing order together with the corresponding permutation matrix $\pi \in \mathbb{K}^{2m\times 2m}$ such that, when $\mathbf{v}$ and $\mathbf{u}$ are seen as column vectors in $\mathbb{Z}_{\geqslant 0}^{2m\times 1}$, we have $\mathbf{v} = \pi\mathbf{u}$. Now that $\mathbf{v}$ is non-decreasing and bounds the corresponding row degree of $\pi\mathbf{Q}$, we compute $\mathbf{N}$ a $\mathbf{v}$-minimal kernel basis for $\pi\mathbf{Q}$ using Algorithm 26, then, $\mathbf{N}\pi$ is a $\mathbf{u}$-minimal kernel basis for $\mathbf{Q}$. Since by assumption $|\mathbf{v}| = |\mathbf{d}| + |\mathbf{t}| \leqslant \xi$, the announced cost bound follows directly from Proposition 12.7.

Finally, we prove the bound on the sum of the $(\mathbf{s}+\mathbf{t})$-row degrees of $\mathbf{R}$. Since $\mathbf{P}$ is $\mathbf{s}$-reduced and $\mathbf{R}$ is $(\mathbf{s}+\mathbf{t})$-reduced, and they are unimodularly equivalent, from item (v) of Theorem 1.11 we have $|\mathrm{rdeg}_{\mathbf{s}+\mathbf{t}}(\mathbf{R})| = \deg(\det(\mathbf{R})) + |\mathbf{s}+\mathbf{t}| = \deg(\det(\mathbf{P})) + |\mathbf{s}| + |\mathbf{t}| = |\mathrm{rdeg}_{\mathbf{d}}(\mathbf{P})| + |\mathbf{t}| = |\mathbf{d}| + |\mathbf{t}|$. ∎

14.2 Computing residuals for interpolant bases

Let $\mathbf{J} \in \mathbb{K}^{D\times D}$ be as in Section 2.4.2; in particular, we suppose that $\mathbf{J}$ is a Jordan matrix, given by a standard representation as in Eq. (2.3). Given a matrix $\mathbf{F} \in \mathbb{K}^{m\times D}$ and a polynomial matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$, we show how to compute the product $\mathbf{P}\cdot\mathbf{F} \in \mathbb{K}^{m\times D}$,

as defined in Section 2.1.2. We will often call the result *residual*, as this is the role this matrix plays in our algorithms for interpolant bases presented in Chapter 13.

We announce here the two main results that we will obtain concerning the computation of these products. The first one concerns the case where one has control over the row degrees of the matrix $\mathbf{P}$, while the second one focuses on the case of controlled column degrees. These are the two situations that we encounter in our interpolant basis algorithms: as explained in Section 1.2.2, having almost uniform shifts implies that the sum of the row degrees is controlled, whereas focusing on shifted Popov bases implies that the sum of the column degrees is controlled. The algorithm JordanMul mentioned in the following proposition will be detailed in Algorithm 32.

Proposition 14.3. *Let $\mathbf{J} \in \mathbb{K}^{D\times D}$ be a Jordan matrix given by a standard representation, let $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$, and let $\mathbf{F} \in \mathbb{K}^{m\times D}$. There is an algorithm* JordanMul *that computes the product $\mathbf{P}\cdot\mathbf{F} \in \mathbb{K}^{m\times D}$; if $D \geqslant m$ and if the sum of the row degrees of $\mathbf{P}$ is in $\mathcal{O}(D)$, this algorithm uses $\mathcal{O}(\mathsf{MM}(m, D/m)\log(D/m) + m\mathsf{M}(D)\log(D))$ operations in $\mathbb{K}$.*

The case of controlled column degrees follows as a consequence; the proof will be given at the end of Section 14.3.

Corollary 14.4. *Let $\mathbf{J} \in \mathbb{K}^{D\times D}$ be a Jordan matrix given by a standard representation, let $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$, and let $\mathbf{F} \in \mathbb{K}^{m\times D}$. If $D \geqslant m$, and if the sum of the column degrees of $\mathbf{P}$ is at most D, then $\mathbf{P}\cdot\mathbf{F}$ can be computed using $\mathcal{O}(\mathsf{MM}(m, D/m)\log(D/m)+m\mathsf{M}(D)\log(D))$ operations in $\mathbb{K}$.*

Remark that when $D \geqslant m$ and the sum of the row or of the column degrees of $\mathbf{P}$ is in $\mathcal{O}(D)$, storing $\mathbf{P}$ requires $\mathcal{O}(mD)$ elements in $\mathbb{K}$, so that representing the input and output of this computation involves $\Theta(mD)$ field elements. At best, one could thus hope for an algorithm of cost $\mathcal{O}(mD)$. Our result is close, as we get a cost of $\mathcal{O}\tilde{}(m^{1.38}D)$ with the best known value of ω.

14.2.1 Residuals and Chinese remaindering

The following lemma writes the output in a more precise manner. The proof is a straightforward consequence of the discussion in Section 2.4.1.

Lemma 14.5. *Suppose that $\mathbf{J}$ has the form $((x_1, D_1), \ldots, (x_n, D_n))$. Let $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ and $\mathbf{F} \in \mathbb{K}^{m\times D}$, and write $\mathbf{F} = [\mathbf{F}_1 \ \cdots \ \mathbf{F}_n]$ with $\mathbf{F}_j$ in $\mathbb{K}^{m\times D_j}$ for $1 \leqslant j \leqslant n$. For $1 \leqslant j \leqslant n$, define the following matrices:*

- *$\mathbf{F}_{j,\text{poly}} = \mathbf{F}_j\,[1, X, \ldots, X^{D_j-1}]^\mathsf{T} \in \mathbb{K}[X]^{m\times 1}$ is the column vector with polynomial entries built from the columns of $\mathbf{F}_j$,*
- *$\mathbf{G}_{j,\text{poly}} = \mathbf{P}(X + x_j)\,\mathbf{F}_{j,\text{poly}} \bmod X^D \in \mathbb{K}[X]^{m\times 1}$,*
- *$\mathbf{G}_j = [\mathbf{G}_{j,0}, \ldots, \mathbf{G}_{j,D_j-1}] \in \mathbb{K}^{m\times D_j}$ is the matrix whose columns are the coefficients of $\mathbf{G}_{j,\text{poly}}$ of degrees $0, \ldots, D_j - 1$.*

Then, $\mathbf{P}\cdot\mathbf{F} = \mathbf{G}$ with $\mathbf{G} = [\mathbf{G}_1 \ \cdots \ \mathbf{G}_n] \in \mathbb{K}^{m\times D}$.

To give an idea of our algorithm's behaviour, let us first consider the case where $\mathbf{J}$ is the upper shift matrix $\mathbf{Z}$ as in Eq. (2.2), that is, there is only one Jordan block whose eigenvalue is 0. This corresponds to having $n = 1$ in the previous lemma, which thus says that we can turn the input $\mathbf{F}$ into a vector of m polynomials of degree at most D, and that we simply have to left-multiply this vector by $\mathbf{P}$. Suppose furthermore that all entries in $\mathbf{P}$ have degree $\mathcal{O}(D/m)$: this is the most natural situation ensuring that the sum of its row degrees is $\mathcal{O}(D)$, as assumed in Proposition 14.3. Then, we have to multiply an $m \times m$ matrix with entries of degree $\mathcal{O}(D/m)$ by an $m \times 1$ vector with entries of degree at most D. To do this efficiently, we have seen in Lemma 6.3 how to rely on partial linearization techniques: we expand the right-hand side into an $m \times m$ polynomial matrix with entries of degree $\mathcal{O}(D/m)$, we multiply it by $\mathbf{P}$, and we recombine the entries of the result. This is thus done using $\mathcal{O}(\mathsf{MM}(m, D/m))$ field operations.

On the other side of the spectrum, we also encountered the case of a *diagonal* matrix $\mathbf{J}$, with diagonal entries $x_1, \ldots, x_D$ (so all D_i's are equal to 1); suppose furthermore that these entries are pairwise distinct. In this case, if $\mathbf{F}_{*,1}, \ldots, \mathbf{F}_{*,D}$ denote the columns of $\mathbf{F}$, Lemma 14.5 shows that the output is the matrix whose columns are $\mathbf{P}(x_1)\mathbf{F}_{*,1}, \ldots, \mathbf{P}(x_D)\mathbf{F}_{*,D}$. Evaluating $\mathbf{P}$ at all x_i's would be too costly, as simply representing all the evaluations requires $m^2 D$ field elements; instead, we interpolate a column vector of m polynomials $[f_1, \ldots, f_m]^\mathsf{T}$ of degree less than D from the rows of $\mathbf{F}$, do the same matrix-vector product as in the previous paragraph, and evaluate the output at the x_i's; the total cost is $\mathcal{O}(\mathsf{MM}(m, D/m) + m\mathsf{M}(D)\log(D))$.

Our main algorithm generalizes these two particular processes. We now state a few basic results that will be needed for this kind of calculation, around problems related to polynomial modular reduction and Chinese remaindering.

Lemma 14.6. *The following cost estimates hold:*

- *Given p of degree d in $\mathbb{K}[X]$, and x in $\mathbb{K}$, one can compute $p(X+x)$ in $\mathcal{O}(\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$.*

- *Given moduli $q_1, \ldots, q_s$ in $\mathbb{K}[X]$, whose sum of degrees is e, and given p of degree $d + e$, then one can compute the modular products $p \bmod q_1, \ldots, p \bmod q_s$ using $\mathcal{O}(\mathsf{M}(d) + \mathsf{M}(e)\log(e))$ operations in $\mathbb{K}$.*

- *Conversely, Chinese remaindering modulo polynomials with sum of degrees d can be done in $\mathcal{O}(\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$.*

Proof. For the first and third point, we refer the reader to [GG13, Chapters 9 and 10]. For the second point, we first compute $q = q_1 \cdots q_s$ in time $\mathcal{O}(\mathsf{M}(e)\log(e))$, reduce p modulo q in time $\mathcal{O}(\mathsf{M}(d+e))$, and use the simultaneous modular reduction algorithm of [GG13, Corollary 10.17], which takes time $\mathcal{O}(\mathsf{M}(e)\log(e))$. We conclude by remarking that $\mathsf{M}(d+e) + \mathsf{M}(e)\log(e) \in \mathcal{O}(\mathsf{M}(d) + \mathsf{M}(e)\log(e))$, as can be seen by considering the cases $d \leqslant e$ and $d > e$. ∎

14.2.2 Main algorithm

For a Jordan matrix $\mathbf{J} \in \mathbb{K}^{D\times D}$ given in standard representation, and for any x in $\mathbb{K}$, we will denote by $\mathrm{rep}(x, \mathbf{J})$ the number of pairs (x, s) appearing in that representation, counting repetitions. In particular, $\sum_{x\in\mathbb{K}} \mathrm{rep}(x, \mathbf{J}) = D$.

For an integer $k \in \{0, \ldots, \lceil \log(D) \rceil\}$, we select from the representation of $\mathbf{J}$ all those pairs (x, s) with s in $\{2^k, \ldots, 2^{k+1} - 1\}$, obtaining a set $\mathbf{J}^{(k)}$. Since $\mathbf{J}$ is in standard representation, we can compute all $\mathbf{J}^{(k)}$ by a single pass through the array $\mathbf{J}$, and we can ensure for free that all $\mathbf{J}^{(k)}$ themselves are in standard representation. We decompose $\mathbf{J}^{(k)}$ further into two classes $\mathbf{J}^{(k,>m)}$, where all pairs (x, s) are such that $\mathrm{rep}(x, \mathbf{J}^{(k)})$ is greater than m, and $\mathbf{J}^{(k,\leqslant m)}$, which contains all other pairs. As above, this decomposition can be done in linear time, and we can ensure for no extra cost that $\mathbf{J}^{(k,>m)}$ and $\mathbf{J}^{(k,\leqslant m)}$ are in standard representation. Explicitly, these sequences will be written as

$$\mathbf{J}^{(k,>m)} = ((x_1^{(k)}, s_{1,1}^{(k)}), \ldots, (x_1^{(k)}, s_{1,r_1^{(k)}}^{(k)}), \ldots, (x_{t^{(k)}}^{(k)}, s_{t^{(k)},1}^{(k)}), \ldots, (x_{t^{(k)}}^{(k)}, s_{t^{(k)},r_{t^{(k)}}^{(k)}}^{(k)})),$$

with $(r_i^{(k)})_i = (\mathrm{rep}(x_i^{(k)}, \mathbf{J}^{(k)}))_i$ non-increasing, and where for i in $\{1, \ldots, t^{(k)}\}$, $r_i^{(k)} > m$ and $(s_{i,j}^{(k)})_j$ is a non-increasing sequence of elements in $\{2^k, \ldots, 2^{k+1} - 1\}$. The corresponding sets of columns in the input matrix $\mathbf{F}$ and the output $\mathbf{G}$ will be written

$$\mathbf{F}^{(k,>m)} = (\mathbf{F}_{i,j}^{(k,>m)})_{1\leqslant i\leqslant t^{(k)}, 1\leqslant j\leqslant r_i^{(k)}}$$

and

$$\mathbf{G}^{(k,>m)} = (\mathbf{G}_{i,j}^{(k,>m)})_{1\leqslant i\leqslant t^{(k)}, 1\leqslant j\leqslant r_i^{(k)}};$$

they will be treated using a direct application of Lemma 14.5. Similarly, we write

$$\mathbf{J}^{(k,\leqslant m)} = ((\xi_1^{(k)}, D_{1,1}^{(k)}), \ldots, (\xi_1^{(k)}, D_{1,\rho_1^{(k)}}^{(k)}), \ldots, (\xi_{\tau^{(k)}}^{(k)}, D_{\tau^{(k)},1}^{(k)}), \ldots, (\xi_{\tau^{(k)}}^{(k)}, D_{\tau^{(k)},\rho_{\tau^{(k)}}^{(k)}}^{(k)})),$$

with $(\rho_i^{(k)})_i = (\mathrm{rep}(\xi_i^{(k)}, \mathbf{J}^{(k)}))_i$ non-increasing, and where for i in $\{1, \ldots, \tau^{(k)}\}$, $\rho_i^{(k)} \leqslant m$ and $(D_{i,j}^{(k)})_j$ is a non-increasing sequence of elements in $\{2^k, \ldots, 2^{k+1} - 1\}$. The corresponding sets of columns in the input matrix $\mathbf{F}$ and the output $\mathbf{G}$ will be written $\mathbf{F}^{(k,\leqslant m)}$ and $\mathbf{G}^{(k,\leqslant m)}$; more precisely, they take the form

$$\mathbf{F}^{(k,\leqslant m)} = (\mathbf{F}_{i,j}^{(k,\leqslant m)})_{1\leqslant i\leqslant \tau^{(k)}, 1\leqslant j\leqslant \rho_i^{(k)}}$$

and

$$\mathbf{G}^{(k,\leqslant m)} = (\mathbf{G}_{i,j}^{(k,\leqslant m)})_{1\leqslant i\leqslant \tau^{(k)}, 1\leqslant j\leqslant \rho_i^{(k)}},$$

and will be treated using a Chinese remaindering approach.

In the main loop, the index k will range from 0 to $\lfloor \log(D/m) \rfloor$. After that stage, all entries (x, s) in $\mathbf{J}$ that were not processed yet are such that $s > D/m$. In particular, if we call $\mathbf{J}^{(\infty,\leqslant m)}$ the set of these remaining entries, we deduce that this set has cardinality at most m; thus $\mathrm{rep}(x, \mathbf{J}^{(\infty,\leqslant m)}) \leqslant m$ holds for all x and we process these entries using the Chinese remaindering approach.

Algorithm JordanMul constructs all these sets $\mathbf{J}^{(k,>m)}$, $\mathbf{J}^{(k,\leqslant m)}$, and $\mathbf{J}^{(\infty,\leqslant m)}$, then extracts the corresponding columns from $\mathbf{F}$ (via the subroutine ExtractColumns), and processes these subsets of columns, before merging all the results.

Algorithm 32 – JordanMul
(Residuals for a multiplication matrix in Jordan form)

Input:

- a Jordan matrix $\mathbf{J}$ in $\mathbb{K}^{D\times D}$ in standard representation,
- a matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$,
- a matrix $\mathbf{F} \in \mathbb{K}^{m\times D}$.

Output: the product $\mathbf{G} = \mathbf{P} \cdot \mathbf{F} \in \mathbb{K}^{m\times D}$.

1. For k from 0 to $\lfloor \log(D/m) \rfloor$
   a. $\mathbf{J}^{(k)} \leftarrow ((x,s) \in \mathbf{J} \mid 2^k \leqslant s < 2^{k+1})$
   b. $\mathbf{J}^{(k,>m)} \leftarrow ((x,s) \in \mathbf{J}^{(k)} \mid \mathrm{rep}(x, \mathbf{J}^{(k)}) > m)$
   c. $\mathbf{F}^{(k,>m)} \leftarrow \mathrm{ExtractColumns}(\mathbf{F}, \mathbf{J}^{(k,>m)})$
   d. $\mathbf{G}^{(k,>m)} \leftarrow \mathrm{JordanMulByShiftingP}(\mathbf{J}^{(k,>m)}, \mathbf{P}, \mathbf{F}^{(k,>m)})$
   e. $\mathbf{J}^{(k,\leqslant m)} \leftarrow ((x,s) \in \mathbf{J}^{(k)} \mid \mathrm{rep}(x, \mathbf{J}^{(k)}) \leqslant m)$
   f. $\mathbf{F}^{(k,\leqslant m)} \leftarrow \mathrm{ExtractColumns}(\mathbf{F}, \mathbf{J}^{(k,\leqslant m)})$
   g. $\mathbf{G}^{(k,\leqslant m)} \leftarrow \mathrm{JordanMulByCRT}(\mathbf{J}^{(k,\leqslant m)}, \mathbf{P}, \mathbf{F}^{(k,\leqslant m)})$
2. $\mathbf{J}^{(\infty,\leqslant m)} \leftarrow ((x,s) \in \mathbf{J} \mid 2^{\lfloor \log(D/m) \rfloor + 1} \leqslant s)$
3. $\mathbf{F}^{(\infty,\leqslant m)} \leftarrow \mathrm{ExtractColumns}(\mathbf{F}, \mathbf{J}^{(\infty,\leqslant m)})$
4. $\mathbf{G}^{(\infty,\leqslant m)} \leftarrow \mathrm{JordanMulByCRT}(\mathbf{J}^{(\infty,\leqslant m)}, \mathbf{P}, \mathbf{F}^{(\infty,\leqslant m)})$
5. Return $\mathrm{Merge}((\mathbf{G}^{(k,>m)})_{0\leqslant k\leqslant \lfloor \log(D/m) \rfloor}, (\mathbf{G}^{(k,\leqslant m)})_{0\leqslant k\leqslant \lfloor \log(D/m) \rfloor}, \mathbf{G}^{(\infty,\leqslant m)})$

14.2.3 Computing the residual by shifting P

We start with the case of the sets $\mathbf{J}^{(k,>m)}$, for which we follow a direct approach. Below, recall that we write

$$\mathbf{J}^{(k,>m)} = ((x_1^{(k)}, s_{1,1}^{(k)}), \ldots, (x_1^{(k)}, s_{1,r_1^{(k)}}^{(k)}), \ldots, (x_{t^{(k)}}^{(k)}, s_{t^{(k)},1}^{(k)}), \ldots, (x_{t^{(k)}}^{(k)}, s_{t^{(k)},r_{t^{(k)}}^{(k)}}^{(k)})),$$

with $s_{i,1}^{(k)} \geqslant s_{i,j}^{(k)}$ for any k, i, and j. For a fixed k, we compute $\mathbf{P}_i^{(k)} = \mathbf{P}(X + x_i^{(k)}) \bmod X^{s_{i,1}^{(k)}}$, for i in $\{1, \ldots, t^{(k)}\}$, and do the corresponding matrix products. This is described in Algorithm 33; we give below a bound on the total time spent in this algorithm, that is, for all k in $\{0, \ldots, \lfloor \log(D/m) \rfloor\}$. Before that, we give two lemmas: the first one will allow us to control the cost of the calculations in this case; in the second one, we explain how to efficiently compute the polynomial matrices $\mathbf{P}_i^{(k)}$.

Lemma 14.7. *The following bound holds:*

$$\sum_{k=0}^{\lfloor \log(D/m) \rfloor} \sum_{i=1}^{t^{(k)}} r_i^{(k)} s_{i,1}^{(k)} \in \mathcal{O}(D).$$

Proof. By construction, we have the estimate

$$\sum_{k=0}^{\lfloor \log(D/m) \rfloor} \sum_{i=1}^{t^{(k)}} \sum_{j=1}^{r_i^{(k)}} s_{i,j}^{(k)} \leqslant D,$$

since this represents the total size of all blocks contained in the sequences $\mathbf{J}^{(k,>m)}$. Now, for fixed k and i, the construction of $\mathbf{J}^{(k)}$ implies that the inequality $s_{i,1}^{(k)} \leqslant 2 s_{i,j}^{(k)}$ holds for all j. This shows that we have

$$r_i^{(k)} s_{i,1}^{(k)} \leqslant 2 \sum_{j=1}^{r_i^{(k)}} s_{i,j}^{(k)},$$

and the conclusion follows by summing over all k and i. ☙

In the following lemma, we explain how to compute the polynomial matrices $\mathbf{P}_i^{(k)}$ in an efficient manner, for i in $\{1, \ldots, t^{(k)}\}$ and for all the values of k we need.

Lemma 14.8. *Suppose that the sum of the row degrees of* $\mathbf{P}$ *is* $\mathcal{O}(D)$. *Then one can compute the matrices* $\mathbf{P}_i^{(k)}$ *for all* k *in* $\{0, \ldots, \lfloor \log(D/m) \rfloor\}$ *and* i *in* $\{1, \ldots, t^{(k)}\}$ *using* $\mathcal{O}(m\, \mathsf{M}(D) \log(D))$ *operations in* $\mathbb{K}$.

Proof. We use the second item in Lemma 14.6 to first compute $\mathbf{P} \bmod (X - x_i^{(k)})^{s_{i,1}^{(k)}}$, for all k and i as in the statement of the lemma. Here, the sum of the degrees is

$$S = \sum_{k,i} s_{i,1}^{(k)},$$

so we get a total cost of $\mathcal{O}(\mathsf{M}(d) + \mathsf{M}(S)\log(S))$ for an entry of $\mathbf{P}$ of degree d. Summing over all entries, and using the fact that the sum of the row degrees of $\mathbf{P}$ is $\mathcal{O}(D)$, we obtain a total cost of

$$\mathcal{O}(m\,\mathsf{M}(D) + m^2\mathsf{M}(S)\log(S)).$$

Now, because we consider here $\mathbf{J}^{(k,>m)}$, we have $r_i^{(k)} > m$ for all k and i. Hence, using the super-linearity of $\mathsf{M}(\cdot)$, the term $m^2\mathsf{M}(S)\log(S)$ admits the upper bound

$$m\,\mathsf{M}\left(\sum_{k,i} r_i^{(k)} s_{i,1}^{(k)}\right)\log(S),$$

which is in $\mathcal{O}(m\,\mathsf{M}(D)\log(D))$ in view of Lemma 14.7.

Then we apply a variable shift to all these polynomials to replace X by $X + x_i^{(k)}$. Using the first item in Lemma 14.6, for fixed k and i, the cost is $\mathcal{O}(m^2\mathsf{M}(s_{i,1}^{(k)})\log(s_{i,1}^{(k)}))$. Hence, the total time is again $\mathcal{O}(m^2\mathsf{M}(S)\log(S))$, so the same overall bound as above holds. ☙

Algorithm 33 – JordanMulByShifting
(Jordan residual via shifting)

Input:

- the Jordan matrix $\mathbf{J}^{(k,>m)}$ given by the standard representation $((x_1^{(k)}, s_{1,1}^{(k)}), \ldots, (x_1^{(k)}, s_{1,r_1^{(k)}}^{(k)}), \ldots, (x_{t^{(k)}}^{(k)}, s_{t^{(k)},1}^{(k)}), \ldots, (x_{t^{(k)}}^{(k)}, s_{t^{(k)},r_{t^{(k)}}^{(k)}}^{(k)}))$,
- a matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$,
- a matrix $\mathbf{F}^{(k,>m)} = [\mathbf{F}_{1,1}^{(k,>m)} \;\cdots\; \mathbf{F}_{t^{(k)},r_{t^{(k)}}^{(k)}}^{(k,>m)}] \in \mathbb{K}^{m\times\sum_{i,j} s_{i,j}^{(k)}}$ with $\mathbf{F}_{i,j}^{(k,>m)} \in \mathbb{K}^{m\times s_{i,j}^{(k)}}$ for all i, j.

Output: the product $\mathbf{P}\cdot\mathbf{F}^{(k,>m)} \in \mathbb{K}^{m\times\sum_{i,j} s_{i,j}^{(k)}}$.

1. $(\mathbf{P}_i^{(k)})_{1\leqslant i\leqslant t^{(k)}} \leftarrow (\mathbf{P}(X + x_i) \bmod X^{s_{i,1}^{(k)}})_{1\leqslant i\leqslant t^{(k)}}$
2. For i from 1 to $t^{(k)}$:
   a. $(\mathbf{F}_{i,j,\mathrm{poly}}^{(k,>m)})_{1\leqslant j\leqslant r_i^{(k)}} \leftarrow (\mathbf{F}_{i,j}^{(k,>m)}[1, X, \ldots, X^{s_{i,j}^{(k)}-1}]^{\mathsf{T}})_{1\leqslant j\leqslant r_i^{(k)}}$
   b. $[\mathbf{G}_{i,1,\mathrm{poly}}^{(k,>m)} \;\cdots\; \mathbf{G}_{i,r_i^{(k)},\mathrm{poly}}^{(k,>m)}] \leftarrow \mathbf{P}_i^{(k)}[\mathbf{F}_{i,1,\mathrm{poly}}^{(k,>m)} \;\cdots\; \mathbf{F}_{i,r_i^{(k)},\mathrm{poly}}^{(k,>m)}]$
   c. For j from 1 to $r_i^{(k)}$: $\mathbf{G}_{i,j}^{(k,>m)} \leftarrow (\mathrm{coeff}(\mathbf{G}_{i,j,\mathrm{poly}}^{(k,>m)}, \ell))_{0\leqslant\ell<s_{i,j}^{(k)}}$
3. Return $[\mathbf{G}_{1,1}^{(k,>m)} \;\cdots\; \mathbf{G}_{t^{(k)},r_{t^{(k)}}^{(k)}}^{(k,>m)}]$

Lemma 14.9. *Algorithm 33 is correct. Given the polynomial matrices computed in Lemma 14.8, the total time spent in this algorithm for all k in $\{0, \ldots, \lfloor\log(D/m)\rfloor\}$ is $\mathcal{O}(\mathsf{MM}(m, D/m))$ operations in $\mathbb{K}$.*

Proof. Correctness of the algorithm follows from Lemma 14.5, so we focus on the cost analysis.

Lemma 14.8 gives the cost of computing all polynomial matrices needed at Step **1**. The only other arithmetic operations are those done in the matrix products at Step **2.b**: we multiply matrices of respective sizes $m \times m$ and $m \times r_i^{(k)}$, with entries of degree less than $s_{i,1}^{(k)}$. For given k and i, since we have $m < r_i^{(k)}$, the cost is $\mathcal{O}(\mathsf{MM}(m, s_{i,1}^{(k)}) r_i^{(k)}/m)$; using the super-linearity $\mathcal{H}_{\mathsf{MM}(\cdot,\cdot)}$ of Eq. (6.2), this is in $\mathcal{O}(\mathsf{MM}(m, r_i^{(k)} s_{i,1}^{(k)}/m))$. Applying again Lemma 14.7, we deduce that the sum over all k and i is $\mathcal{O}(\mathsf{MM}(m, D/m))$. ☙

14.2.4 Computing the residual by Chinese remaindering

The second case to consider is $\mathbf{J}^{(k,\leqslant m)}$. Recall that for a given index k, we write this sequence as

$$\mathbf{J}^{(k,\leqslant m)} = ((\xi_1^{(k)}, D_{1,1}^{(k)}), \ldots, (\xi_1^{(k)}, D_{1,\rho_1^{(k)}}^{(k)}), \ldots, (\xi_{\tau^{(k)}}^{(k)}, D_{\tau^{(k)},1}^{(k)}), \ldots, (\xi_{\tau^{(k)}}^{(k)}, D_{\tau^{(k)},\rho_{\tau^{(k)}}^{(k)}}^{(k)})),$$

with $\rho_{\tau^{(k)}}^{(k)} \leqslant \cdots \leqslant \rho_1^{(k)} \leqslant m$ for all i in $\{1, \ldots, \tau^{(k)}\}$. In this case, $\tau^{(k)}$ may be large so the previous approach may lead us to compute too many matrices $\mathbf{P}_i^{(k)}$. Instead, for fixed k and j, we use Chinese remaindering to transform the corresponding submatrices $\mathbf{F}_{i,j}^{(k,\leqslant m)}$ into a polynomial matrix $\mathbf{F}_j^{(k,\leqslant m)}$ of small column dimension; this allows us to efficiently perform matrix multiplication by $\mathbf{P}$ on the left, and we eventually get $\mathbf{P} \cdot \mathbf{F}_{i,j}^{(k,\leqslant m)}$ by computing the first coefficients in a Taylor expansion of this product around every $\xi_i^{(k)}$.

To simplify the notation in the algorithm, we also suppose that for a fixed k, the points $\xi_1^{(k)}, \ldots, \xi_{\tau^{(k)}}^{(k)}$ all appear the same number of times in $\mathbf{J}^{(k,\leqslant m)}$. This is done by replacing $\rho_1^{(k)}, \ldots, \rho_{\tau^{(k)}}^{(k)}$ by their maximum $\rho_1^{(k)}$ (simply written $\rho^{(k)}$ in the pseudo-code) and adding suitable blocks $(\xi_i^{(k)}, D_{i,j}^{(k)})$, with all new $D_{i,j}^{(k)}$ set to zero.

Lemma 14.10. *Algorithm 34 is correct. If the sum of the row degrees of* $\mathbf{P}$ *is in* $\mathcal{O}(D)$*, the total time spent in this algorithm for all* k *in* $\{0, \ldots, \lfloor \log(D/m) \rfloor, \infty\}$ *is*

$$\mathcal{O}(\mathsf{MM}(m, D/m) \log(D/m) + m\mathsf{M}(D) \log(D))$$

operations in $\mathbb{K}$.

Proof. Proving correctness amounts to verifying that we compute the quantities described in Lemma 14.5. Indeed, the formulas in the algorithm show that for all k, i, j, we have $\mathbf{G}_{i,j,\text{shifted}}^{(k,\leqslant m)} = \mathbf{P}\, \mathbf{F}_{i,j,\text{shifted}}^{(k,\leqslant m)} \bmod (X - \xi_i^{(k)})^{D_{i,j}^{(k)}}$; the link with Lemma 14.5 is made by observing that $\mathbf{F}_{i,j,\text{shifted}}^{(k,\leqslant m)} = \mathbf{F}_{i,j,\text{poly}}^{(k,\leqslant m)}(X - \xi_i^{(k)})$ and $\mathbf{G}_{i,j,\text{shifted}}^{(k,\leqslant m)} = \mathbf{G}_{i,j,\text{poly}}^{(k,\leqslant m)}(X - \xi_i^{(k)})$.

In terms of complexity, the first item in Lemma 14.6 shows that for a given index k, Step **1.a** can be done in time

$$\mathcal{O}\left(m \sum_{i,j} \mathsf{M}\left(D_{i,j}^{(k)}\right) \log\left(D_{i,j}^{(k)}\right) \right),$$

Algorithm 34 – JordanMulByCRT
(Jordan residual via Chinese remaindering)

Input:

- the Jordan matrix $\mathbf{J}^{(k,\leqslant m)}$ given by the standard representation $((\xi_1^{(k)}, D_{1,1}^{(k)}), \ldots, (\xi_1^{(k)}, D_{1,\rho^{(k)}}^{(k)}), \ldots, (\xi_{\tau^{(k)}}^{(k)}, D_{\tau^{(k)},1}^{(k)}), \ldots, (\xi_{\tau^{(k)}}^{(k)}, D_{\tau^{(k)},\rho^{(k)}}^{(k)}))$,
- a matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$,
- a matrix $\mathbf{F}^{(k,\leqslant m)} = [\mathbf{F}_{1,1}^{(k,\leqslant m)} \quad \cdots \quad \mathbf{F}_{\tau^{(k)},\rho^{(k)}}^{(k,\leqslant m)}] \in \mathbb{K}^{m\times \sum_{i,j} D_{i,j}^{(k)}}$ with $\mathbf{F}_{i,j}^{(k,\leqslant m)} \in \mathbb{K}^{m\times D_{i,j}^{(k)}}$ for all i, j.

Output: the product $\mathbf{P}\cdot\mathbf{F}^{(k,\leqslant m)} \in \mathbb{K}^{m\times \sum_{i,j} D_{i,j}^{(k)}}$.

1. For j from 1 to $\rho^{(k)}$:
   a. $(\mathbf{F}_{i,j,\text{shifted}}^{(k,\leqslant m)})_{1\leqslant i\leqslant \tau^{(k)}} \leftarrow (\mathbf{F}_{i,j}^{(k,\leqslant m)}[1, X-\xi_i^{(k)}, \ldots, (X-\xi_i^{(k)})^{D_{i,j}^{(k)}-1}]^{\mathsf{T}})_{1\leqslant i\leqslant \tau^{(k)}}$
   b. $\mathbf{F}_{j,\text{shifted}}^{(k,\leqslant m)} \leftarrow \mathrm{CRT}((\mathbf{F}_{i,j,\text{shifted}}^{(k,\leqslant m)})_{1\leqslant i\leqslant \tau^{(k)}}, ((X-\xi_i^{(k)})^{D_{i,j}^{(k)}})_{1\leqslant i\leqslant \tau^{(k)}})$
2. $[\mathbf{G}_{1,\text{shifted}}^{(k,\leqslant m)} \quad \cdots \quad \mathbf{G}_{\rho^{(k)},\text{shifted}}^{(k,\leqslant m)}] \leftarrow \mathbf{P}[\mathbf{F}_{1,\text{shifted}}^{(k,\leqslant m)} \quad \cdots \quad \mathbf{F}_{\rho^{(k)},\text{shifted}}^{(k,\leqslant m)}]$
3. For j from 1 to $\rho^{(k)}$
   a. $(\mathbf{G}_{i,j,\text{shifted}}^{(k,\leqslant m)})_{1\leqslant i\leqslant \tau^{(k)}} \leftarrow (\mathbf{G}_{j,\text{shifted}}^{(k,\leqslant m)} \bmod (X-\xi_i^{(k)})^{D_{i,j}^{(k)}})_{1\leqslant i\leqslant \tau^{(k)}}$
   b. $\mathbf{G}_{i,j}^{(k,\leqslant m)} \leftarrow (\mathrm{coeff}(\mathbf{G}_{i,j,\text{shifted}}^{(k,\leqslant m)}(X+\xi_i^{(k,\leqslant m)}), \ell))_{0\leqslant \ell < D_{i,j}^{(k)}}$
4. Return $[\mathbf{G}_{1,1}^{(k,\leqslant m)} \quad \cdots \quad \mathbf{G}_{\tau^{(k)},\rho^{(k)}}^{(k,\leqslant m)}]$

for a total cost of $\mathcal{O}(m\,\mathsf{M}(D)\log(D))$. Step **1.b** can be done in quasi-linear time as well: for each k and j, we can compute each of the m entries of the polynomial vector $\mathbf{F}^{(k,\leqslant m)}_{j,\text{shifted}}$ by fast Chinese remaindering (third item in Lemma 14.6), using

$$\mathcal{O}\Big(\mathsf{M}\Big(S_j^{(k)}\Big)\log\Big(S_j^{(k)}\Big)\Big)$$

operations in $\mathbb{K}$, with $S_j^{(k)} = \sum_i D_{i,j}^{(k)}$. Taking all rows into account, and summing over all indices k and j, we obtain again a total cost of $\mathcal{O}(m\,\mathsf{M}(D)\log(D))$.

The next step to examine is the polynomial matrix product at Step **2**. The matrix $\mathbf{P}$ has size $m \times m$, and the sum of its row degrees is by assumption $\mathcal{O}(D)$; using the partial linearization technique presented in Section 12.2, we can replace $\mathbf{P}$ by a matrix of size $\mathcal{O}(m) \times m$ with entries of degree at most D/m.

For a fixed choice of k, the right-hand side has size $m \times \rho^{(k)}$, and its columns have respective degrees less than $S_1^{(k)}, \ldots, S_{\rho^{(k)}}^{(k)}$. We split each of its columns into new columns of degree at most D/m, so that the jth column is split into $\mathcal{O}(1 + S_j^{(k)}m/D)$ columns (the constant term 1 dominates when $S_j^{(k)} \leqslant D/m$). Thus, the new right-hand side has $\mathcal{O}(\rho^{(k)} + (S_1^{(k)} + \cdots + S_{\rho^{(k)}}^{(k)})m/D)$ columns and degree at most D/m.

Now, taking all k into account, we remark that the left-hand side remains the same; thus, we are led to do one matrix product with degrees D/m, with left-hand side of size $\mathcal{O}(m) \times m$, and right-hand side having column dimension at most

$$\sum_{k\in\{0,\ldots,\lfloor\log(D/m)\rfloor\}\cup\{\infty\}} \rho^{(k)} + \frac{(S_1^{(k)} + \cdots + S_{\rho^{(k)}}^{(k)})m}{D}.$$

Since all $\rho^{(k)}$ are at most m, the first term sums up to $\mathcal{O}(m\log(D/m))$; by construction, the second one adds up to $\mathcal{O}(m)$. Hence, by the super-linearity property $\mathcal{H}_{\mathsf{MM}(\cdot,\cdot)}$ of Eq. (6.2), the matrix product we are considering can be done in time $\mathcal{O}(\mathsf{MM}(m, D/m)\log(D/m))$.

For a given k, $\mathbf{G}^{(k,\leqslant m)}_{1,\text{shifted}}, \ldots, \mathbf{G}^{(k,\leqslant m)}_{\rho^{(k)},\text{shifted}}$ are vectors of size m. Furthermore, for each j the entries of $\mathbf{G}^{(k,\leqslant m)}_{j,\text{shifted}}$ have degree less than $S_1^{(k)} + d_1, \ldots, S_m^{(k)} + d_m$ respectively, where $d_1, \ldots, d_m$ are the degrees of the rows of $\mathbf{P}$. In particular, for a fixed k, the reductions at Step **3.a** can be done in time

$$\mathcal{O}\left(\rho^{(k)}(\mathsf{M}(d_1 + \cdots + d_m)) + m\sum_{j=1}^{\rho^{(k)}} \mathsf{M}(S_j^{(k)})\log(S_j^{(k)})\right)$$

using fast multiple reduction, by means of the second item in Lemma 14.6. Using our assumption on $\mathbf{P}$, and the fact that $\rho^{(k)} \leqslant m$, we see that the first term is $\mathcal{O}(m\mathsf{M}(D))$, which adds up to $\mathcal{O}(m\mathsf{M}(D)\log(D/m))$ if we sum over k. The second term adds up to $\mathcal{O}(m\mathsf{M}(D)\log(D))$, as was the case for Step **1.b**.

The same analysis is used for the shifts taking place at Step **3.b** as for those in Step **1.a**: for fixed k and j, the cost is $\mathcal{O}(m\mathsf{M}(S_j^{(k)})\log(S_j^{(k)}))$, and we conclude as above. ☙

14.3 Computing interpolant bases with known minimal degree

In this section, we design a fast algorithm for computing the shifted Popov interpolant basis when the shifted minimal degree is known, relying on our fast minimal interpolant basis algorithm for almost uniform shift.

Let $\mathbf{J}$ be a Jordan matrix given by a standard representation, let $\mathbf{F} \in \mathbb{K}^{m\times D}$, let $\mathbf{s} \in \mathbb{Z}^m$, and let $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$ denote the $\mathbf{s}$-minimal degree of $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$. Then, denote by $\mathbf{P}$ the $\mathbf{s}$-Popov interpolant basis of $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$; we suppose that $\boldsymbol{\delta}$ is known a priori, and we want to find $\mathbf{P}$.

We recall from Section 1.2.1 that the shift $\mathbf{d} = -\boldsymbol{\delta}$ leads to the same $\mathbf{d}$-Popov interpolant basis $\mathbf{P}$ as the initial shift $\mathbf{s}$, and furthermore that $\mathbf{P}$ can be easily recovered from any interpolant basis which is simply $\mathbf{d}$-reduced.

More precisely, Lemma 1.26 indicates that any $\mathbf{d}$-minimal interpolant basis $\mathbf{R}$ for $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$ has column degree $\boldsymbol{\delta}$, and thus size at most $m^2 + m|\boldsymbol{\delta}|$, which for $D \geqslant m$ is in $\mathcal{O}(mD)$. Yet, the fast algorithm in Section 13.2 for almost uniform shifts cannot directly be used to compute such an $\mathbf{R}$ efficiently, since $|\mathbf{d} - \min(\mathbf{d})|$ can be as large as $\Theta(mD)$, for example when $\boldsymbol{\delta} = (D, 0, \ldots, 0)$. In this case, according to Proposition 13.3, Algorithm 29 uses $\tilde{\mathcal{O}}(m^\omega D)$ operations in $\mathbb{K}$ to find $\mathbf{R}$.

However, by Lemma 2.10, $\mathbf{d} = -\boldsymbol{\delta}$ satisfies

$$|\max(\mathbf{d}) - \mathbf{d}| \leqslant |\boldsymbol{\delta}| = \deg(\det(\mathbf{P})) \leqslant D.$$

In other words, the shift $\mathbf{d} = -\boldsymbol{\delta}$ is almost uniform around its maximum value; notice that Algorithm 29 is efficient for shifts that are almost uniform around their minimum value. We solve this difficulty by relying on the partial linearization technique detailed in Section 6.2 and Lemma 6.2, and used before in Algorithms 14 and 18 for computing approximant bases in solution bases when the shifted minimal degree is known. This leads us to Algorithm 35.

Proposition 14.11. *Algorithm 35 is correct, and if $D \geqslant m$ it uses*

$$\begin{aligned}&\mathcal{O}(m^{\omega-1}\mathsf{M}(D) + m^\omega \mathsf{M}(D/m)\log(D/m)^2 + m^{\omega-1}D\log(m) + m\mathsf{M}(D)\log(D)\log(D/m))\\ &\quad\subseteq \mathcal{O}(m^{\omega-1}\mathsf{M}(D)\log(D)\log(D/m))\end{aligned}$$

operations in $\mathbb{K}$.

Proof. We focus on the case $D \geqslant m$; otherwise, a better cost bound can be achieved even without knowing $\boldsymbol{\delta}$ using dense linear algebra (see Proposition 4.18). The correctness of the algorithm follows from Lemma 6.2.

Let us now prove the cost bound. First, according to Lemma 6.2 we have $\max(\mathbf{d}) - \min(\mathbf{d}) \leqslant \lceil D/m \rceil$, hence $|\mathbf{d} - \min(\mathbf{d})| \in \mathcal{O}(D)$. Then, from Proposition 13.3, we directly obtain that Step **4** is done within the above cost bound.

Besides, Step **3** can be done in $\mathcal{O}(m\mathsf{M}(D)\log(D))$ field operations via Algorithm 32 for computing residuals, as explained in Lemma 14.12 below.

Algorithm 35 – MinDegIntBas
(Interpolant basis with known minimal degree)

Input:

- a Jordan matrix $\mathbf{J} \in \mathbb{K}^{D \times D}$ in standard representation,
- a matrix $\mathbf{F} \in \mathbb{K}^{m \times D}$,
- a shift $\mathbf{s} \in \mathbb{Z}^m$,
- the $\mathbf{s}$-minimal degree $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m) \in \mathbb{Z}^m_{\geqslant 0}$ of $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$.

Output: the $\mathbf{s}$-Popov interpolant basis of $\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$.

1. $\delta \leftarrow \lceil D/m \rceil$,
   $\alpha_i \leftarrow \max(1, \lceil \delta_i/\delta \rceil)$ for $1 \leqslant i \leqslant m$,
   $\widetilde{m} \leftarrow \alpha_1 + \cdots + \alpha_m$
2. $\widetilde{\boldsymbol{\delta}} \in \mathbb{Z}^{\widetilde{m}}_{\geqslant 0}$ as in Eq. (6.4)
   $\mathbf{d} \leftarrow -\widetilde{\boldsymbol{\delta}} \in \mathbb{Z}^{\widetilde{m}}_{\geqslant 0}$
3. $\mathcal{E} \in \mathbb{K}[X]^{\widetilde{m} \times m}$ as in Eq. (6.5)
   $\widetilde{\mathbf{F}} \leftarrow \mathcal{E} \cdot \mathbf{F}$ `// using` JordanMul
4. $\mathbf{R} \leftarrow$ ShiftMinIntBas$(\widetilde{\mathbf{F}}, \mathbf{J}, \mathbf{d})$
5. Return the submatrix of $\mathrm{lm}_{\mathbf{d}}(\mathbf{R})^{-1}\mathbf{R}\mathcal{E}$ formed by the rows at indices $\alpha_1 + \cdots + \alpha_i$ for $1 \leqslant i \leqslant m$

Lemma 1.26 proves that the sum of the column degrees of $\mathbf{R}$ is $|\widetilde{\boldsymbol{\delta}}|$, which is equal to $|\boldsymbol{\delta}|$ by construction (see Eq. (6.4)). Then, Lemma 2.10 implies that $|\widetilde{\boldsymbol{\delta}}| \leqslant D$. As a consequence, the product $\mathrm{lm}_{\mathbf{d}}(\mathbf{R})^{-1}\mathbf{R}$ at Step **5** can be done in $\mathcal{O}(m^{\omega-1}D)$ operations, by first linearizing the columns of $\mathbf{R}$ into a $\widetilde{m} \times (\widetilde{m} + |\widetilde{\boldsymbol{\delta}}|)$ matrix over $\mathbb{K}$, then left-multiplying this matrix by $\mathrm{lm}_{\mathbf{d}}(\mathbf{R})^{-1}$ (itself computed using $\mathcal{O}(m^{\omega})$ operations), and finally compressing back the columns to obtain the result. Because of the degrees in $\widetilde{\mathbf{P}}$ and the definition of $\mathcal{E}$ (see Eq. (6.5)), the output in Step **5** can be formed from this product without using any arithmetic operation. ☙

The efficient computation of $\mathcal{E} \cdot \mathbf{F}$ can be done with the algorithm JordanMul for computing residuals detailed in Section 14.2.

Lemma 14.12. *The product $\mathcal{E} \cdot \mathbf{F}$ at Step **3** of Algorithm 35 can be computed using $\mathcal{O}(m\mathsf{M}(D)\log(D))$ operations in $\mathbb{K}$.*

Proof. This matrix $\mathcal{E} \cdot \mathbf{F}$ has $\widetilde{m}$ rows, with $\widetilde{m} \leqslant 2m$ according to Lemma 6.2. Besides, by definition of $\mathcal{E}$, each row of $\mathcal{E} \cdot \mathbf{F}$ is a product of the form $X^{i\delta} \cdot \mathbf{F}_{j,*}$, where $0 \leqslant i \leqslant m$, $1 \leqslant j \leqslant m$, $\mathbf{F}_{j,*}$ denotes the row j of $\mathbf{F}$, and $\delta = \lceil D/m \rceil$ as in Lemma 6.2. In particular, $i\delta \leqslant 2D$: then, according to Proposition 14.3, each of these $\widetilde{m}$ products can be computed using $\mathcal{O}(\mathsf{M}(D)\log(D))$ operations in $\mathbb{K}$. ☙

This lemma and the partial linearization technique in Lemma 6.2 can also be used to compute the residual at Step **2.c** of Algorithm 30, that is, a product of the form $\mathbf{P} \cdot \mathbf{F}$ with the sum of the column degrees of $\mathbf{P}$ bounded by D. First, we expand the high-degree columns of $\mathbf{P}$ to obtain $\widetilde{\mathbf{P}} \in \mathbb{K}[X]^{m \times \widetilde{m}}$ of degree less than $\lceil D/m \rceil$ such that $\mathbf{P} = \widetilde{\mathbf{P}}\mathcal{E}$; then, we compute $\widetilde{\mathbf{F}} = \mathcal{E} \cdot \mathbf{F}$; and finally we rely on the algorithm supporting Proposition 14.3 to compute $\mathbf{P} \cdot \mathbf{F} = \widetilde{\mathbf{P}} \cdot \widetilde{\mathbf{F}}$ efficiently. This proves Corollary 14.4.

Part V

Normal forms of polynomial matrices

Contents

Chapter 15 Shifted Popov forms **285**

15.1 The generic determinant degree bound 285

15.2 Reducing to almost uniform input degrees 286

15.2.1 Column partial linearization 287

15.2.2 Row partial linearization 291

15.2.3 Reducing the degrees in shifted Popov form computation 292

15.3 Fast, probabilistic computation of the shifted Popov form 293

Chapter 16 Hermite form and determinant **297**

16.1 Preliminaries: column bases 298

16.2 Computing the diagonal entries of a triangular form 299

16.2.1 Fast block elimination 300

16.2.2 Computational cost and example 300

16.3 Fast computation of the determinant of a polynomial matrix 303

16.4 Fast Hermite form algorithm with known minimal degree 309

16.4.1 Hermite form via shifted column reduction 309

16.4.2 Reducing the amplitude of the minimal degree 310

16.4.3 Algorithm and computational cost 314

16.4.4 Proof of Lemma 16.19 317

16.5 Reduction to almost uniform input degrees 319

15

Shifted Popov forms

In this chapter, we consider the problem of computing the shifted Popov form of a nonsingular polynomial matrix $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$ (Problem 15 and Section 3.2).

After explaining our interest in cost bounds involving the generic determinant bound $D_{\mathbf{A}}$, we give the details of how we exploit partial linearization [GSSV12] to reduce the degree of the input matrix to being at most $D_{\mathbf{A}}$ (Theorem 3.5). As a consequence, any algorithm with cost bound $\mathcal{O}\tilde{}(m^\omega \deg(\mathbf{A}))$ for Problem 15 can be modified to obtain the cost bound $\mathcal{O}\tilde{}(m^\omega \lceil D_{\mathbf{A}}/m \rceil)$.

Then, we show how Smith form computations leads to reducing shifted Popov form computation to systems of linear modular equations. This reduction is efficient thanks to algorithms in [Sto03, Gup11], leading to the first algorithm which computes the shifted Popov form of $\mathbf{A}$ in $\mathcal{O}\tilde{}(m^\omega \deg(\mathbf{A}))$ for an arbitrary shift.

15.1 The generic determinant degree bound

Here, we recall the definition of the *generic determinant bound* [GSSV12]. Concerning the computation of normal forms of polynomial matrices, we also detail why we are interested in having cost bounds that are quasi-linear in this parameter, rather than in the degree or the average column degree of the input matrix.

For a nonsingular $m \times m$ matrix $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$, the degree of the determinant of $\mathbf{A}$ provides a good measure of the size of the output $\mathbf{P}$ in the case of shifted Popov form computation. Indeed, if we denote by $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m)$ the degrees of the diagonal entries of $\mathbf{P}$, then we have $\deg(\det(\mathbf{A})) = \deg(\det(\mathbf{P})) = \delta_1 + \cdots + \delta_m$. In what follows, we write $|\boldsymbol{\delta}|$ for the sum of the entries of a tuple $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$ with nonnegative entries. Since the diagonal entries are those of largest degree in their respective columns, we directly obtain that $\mathbf{P}$ can be represented using $m^2 + m|\boldsymbol{\delta}| = m^2 + m\deg(\det(\mathbf{A}))$ field elements.

The size of the input $\mathbf{A}$ can be measured in several ways; those below all involve an upper bound on $\deg(\det(\mathbf{A}))$. A first, coarse bound is given by the maximum degree of the entries of the matrix: $\mathbf{A}$ can be represented by $m^2+m^2\deg(\mathbf{A})$ field elements. On the other hand, by definition of the determinant we have that $\deg(\det(\mathbf{A})) \leqslant m\deg(\mathbf{A})$. A second, finer bound can be obtained using the *average* of the row degrees and of the column degrees: the size of $\mathbf{A}$ in terms of field elements is at most $m^2+m\min(|\mathrm{rdeg}(\mathbf{A})|, |\mathrm{cdeg}(\mathbf{A})|)$;

again we have the related bound

$$\deg(\det(\mathbf{A})) \leqslant \min(|\mathrm{rdeg}(\mathbf{A})|, |\mathrm{cdeg}(\mathbf{A})|).$$

An even finer bound on the size of $\mathbf{A}$ is given by the *generic determinant bound*. This notion was introduced in [GSSV12, Section 6] in conjunction with partial linearization for the purpose of managing the non-uniformity of the degrees in the manipulated matrices. For $\mathbf{A} = [a_{i,j}]_{i,j} \in \mathbb{K}[X]^{m\times m}$, this quantity is defined as

$$D_{\mathbf{A}} = \max_{\pi\in S_m} \sum_{1\leqslant i\leqslant m} \overline{\deg}(a_{i,\pi_i}) \tag{15.1}$$

where S_m is the set of permutations of $\{1,\dots,m\}$, and where

$$\overline{\deg}(p) = \begin{cases} 0 & \text{if} \quad p = 0 \\ \deg(p) & \text{if} \quad p \neq 0 \end{cases}.$$

By definition, we have the inequalities

$$\deg(\det(\mathbf{A})) \leqslant D_{\mathbf{A}} \leqslant \min(|\mathrm{rdeg}(\mathbf{A})|, |\mathrm{cdeg}(\mathbf{A})|) \leqslant m\deg(\mathbf{A}),$$

and it is easily verified that $\mathbf{A}$ can be represented using $m^2 + 2mD_{\mathbf{A}}$ field elements.

Thus, in shifted Popov form computation as in Problem 15, both the input and the output have average degree in $\mathcal{O}(D_{\mathbf{A}}/m)$ and can be represented using $\mathcal{O}(m^2\lceil D_{\mathbf{A}}/m\rceil)$ field elements. Furthermore $D_{\mathbf{A}}$ gives a more precise account of the degrees in $\mathbf{A}$ than the average row and column degrees, and an algorithm with cost bound $\tilde{\mathcal{O}}(m^\omega\lceil D_{\mathbf{A}}/m\rceil)$ is always faster, sometimes significantly, than an algorithm with cost bound $\tilde{\mathcal{O}}(m^\omega\lceil s\rceil)$ where s is the average column degree or the average row degree, let alone $s = \deg(\mathbf{A})$.

Remark 15.1. Let us justify why this can sometimes be *significantly* faster. We have seen that $D_{\mathbf{A}}/m$ is bounded from above by both the average column degree and the average row degree of $\mathbf{A}$. It turns out that, in some important cases $D_{\mathbf{A}}/m$ may be substantially smaller than these averages. For example, consider $\mathbf{A}$ with one row and one column of uniformly large degree d and all other entries of degree 0:

$$\mathbf{A} = \begin{bmatrix} [d] & [d] & \cdots & [d] \\ [d] & [0] & \cdots & [0] \\ \vdots & \vdots & \ddots & \vdots \\ [d] & [0] & \cdots & [0] \end{bmatrix} \in \mathbb{K}[X]^{m\times m}.$$

In this example the average row degree and the average column degree are both exactly d while the generic determinant bound is d as well. Here, $D_{\mathbf{A}}/m = d/m$ is much smaller than $d = \deg(\mathbf{A}) = |\mathrm{rdeg}(\mathbf{A})|/m = |\mathrm{cdeg}(\mathbf{A})|/m$. ☕

15.2 Reducing to almost uniform input degrees

We have explained in Section 15.1 our interest in obtaining a cost bound which involves the generic determinant bound, to take into account the fact that the degrees of the input

matrix are possibly unbalanced. The goal of this section is to give a reduction from the general case of shifted Popov form computation to the case where the degree of the input matrix $\mathbf{A}$ is in $\mathcal{O}(\lceil D_{\mathbf{A}}/m \rceil)$. For this, we make use of the partial linearization techniques in [GSSV12, Section 6], leading to Theorem 3.5.

To get a rough idea of how partial linearization works and how it benefits normal form computation, consider the matrix

$$\mathbf{A} = \begin{bmatrix} 1 & X^{39} + X \\ X & X^{41} + 1 \end{bmatrix}.$$

In this case the column degrees of the matrix are quite unbalanced as 1 and 41 have an average column degree of 21. However we can create a second matrix, of slightly larger dimension, as

$$\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A}) = \begin{bmatrix} 1 & X & X^{17} \\ X & 1 & X^{19} \\ 0 & -X^{22} & 1 \end{bmatrix}$$

which shares some nice properties with $\mathbf{A}$ (here, the superscript "c" is used to indicate that we are doing *column* partial linearization). This matrix is constructed by dividing the second column into its two X^{22}-adic coefficients (rows 1 and 2) and then including an additional row (row 3) which provides the single column operation which would undo the division. Thus by construction this matrix is left-unimodularly equivalent to

$$\begin{bmatrix} 1 & X^{39} + X & 0 \\ X & X^{41} + 1 & 0 \\ 0 & -X^{22} & 1 \end{bmatrix}$$

and it is easily seen that the Hermite form of $\mathbf{A}$ will be given by the 2×2 leading submatrix of the Hermite form of $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$. As such we rely on the computation of the Hermite form of a matrix, not much larger than the original matrix, but having the nice property that the degrees are much more uniformly distributed.

15.2.1 Column partial linearization

Column partial linearization of $\mathbf{A}$ transforms the columns of large degree into several columns of lower degree, and for each new column it also augments $\mathbf{A}$ with some type of elementary rows. The latter rows allow us to preserve properties of the matrix, including for example its determinant, its Smith form, and most importantly for us its shifted Popov form, up to the use of a well-chosen shift. We first describe the elementary rows. In the definition below, $\boldsymbol{\delta}$ is typically, but not necessarily, the column degree of $\mathbf{A}$.

Definition 15.2. *Let $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m) \in \mathbb{Z}^m_{\geqslant 0}$ and $\delta = 1 + \lfloor (\delta_1 + \cdots + \delta_m)/m \rfloor$. For each $i \in \{1, \ldots, m\}$, write $\delta_i = (\alpha_i - 1)\delta + \beta_i$ with $\alpha_i \geqslant 1$ and $0 \leqslant \beta_i < \delta$, and let*

$\widetilde{m} = \alpha_1 + \cdots + \alpha_m$. *Define the expansion-compression matrix* $\mathcal{E} \in \mathbb{K}[X]^{\widetilde{m}\times m}$ *as*

$$\mathcal{E} = \begin{bmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ X^{\delta} & & & \\ \vdots & & & \\ X^{(\alpha_1-1)\delta} & & & \\ & \ddots & & \\ & & X^{\delta} & \\ & & \vdots & \\ & & X^{(\alpha_m-1)\delta} & \end{bmatrix} \tag{15.2}$$

and the matrix $\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}} \in \mathbb{K}[X]^{(\widetilde{m}-m)\times\widetilde{m}}$ *as follows:*

(i) for $1 \leqslant i \leqslant m$ *such that* $\alpha_i > 1$, *the row* $(\alpha_1 - 1) + \cdots + (\alpha_{i-1} - 1) + 1$ *of* $\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}}$ *is*

$$\begin{bmatrix} 0 & \cdots & 0 & -X^{\delta} & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{bmatrix}$$

with the entry 1 *at index* $m + (\alpha_1 - 1) + \cdots + (\alpha_{i-1} - 1) + 1$ *and the entry* $-X^{\delta}$ *at index* i,

(ii) for $1 \leqslant i \leqslant m$ *and* $2 \leqslant j \leqslant \alpha_i - 1$, *the row* $(\alpha_1 - 1) + \cdots + (\alpha_{i-1} - 1) + j$ *of* $\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}}$ *is*

$$\begin{bmatrix} 0 & \cdots & 0 & -X^{\delta} & 1 & 0 & \cdots & 0 \end{bmatrix}$$

with the entry 1 *at index* $m + (\alpha_1 - 1) + \cdots + (\alpha_{i-1} - 1) + j$.

By construction the $(\widetilde{m} - m) \times (\widetilde{m} - m)$ submatrix of $\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}}$ formed by its rightmost $\widetilde{m} - m$ columns is lower triangular with 1 on the diagonal, and hence is unimodular. This will play an important role in the properties of the partial linearization of $\mathbf{A}$. Notice also that $\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}}\,\mathcal{E} = \mathbf{0}$.

The column partial linearization of a matrix $\mathbf{A}$ is formed by this block $\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}}$ of elementary rows, and by another block of m rows containing the expanded columns of $\mathbf{A}$. The i-th column of $\mathbf{A}$ is expanded into α_i columns, all having degree at most δ except possibly the one which contains the highest degree entries, whose index is denoted by ρ_i.

Definition 15.3. *Let* $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$ *and* $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m) \in \mathbb{Z}^m_{\geqslant 0}$. *Using the notation in Definition 15.2, let* $\rho_1, \ldots, \rho_m$ *with*

$$\rho_i = \begin{cases} i & \text{if } \alpha_i = 1, \\ m + (\alpha_1 - 1) + \cdots + (\alpha_i - 1) & \text{if } \alpha_i > 1. \end{cases} \tag{15.3}$$

The column partial linearization of $\mathbf{A}$ *is the matrix* $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A}) \in \mathbb{K}[X]^{\widetilde{m}\times\widetilde{m}}$ *defined by:*

(i) the last $\widetilde{m} - m$ *rows of* $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$ *are* $\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}}$,

(ii) the first m rows of $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$ form the unique matrix $\widetilde{\mathbf{A}} \in \mathbb{K}[X]^{m\times\widetilde{m}}$ such that $\mathbf{A} = \widetilde{\mathbf{A}}\mathcal{E}$ and $\widetilde{\mathbf{A}}$ has all columns of degree less than δ except possibly those at indices $\rho_1, \ldots, \rho_m$.

Example 15.4 (Column partial linearization). Let $\mathbb{K}$ be the finite field with 97 elements. Using a computer algebra system, we choose $\mathbf{A} \in \mathbb{K}[X]^{4\times 4}$ with prescribed degrees and random coefficients from $\mathbb{K}$. Instead of showing the entire matrix let us only consider the degree profile which in this case is

$$\mathbf{A} = \begin{bmatrix} [2] & [10] & [63] & [5] \\ [75] & [51] & [95] & [69] \\ [4] & [5] & [48] & [7] \\ [10] & [54] & [75] & [6] \end{bmatrix},$$

where $[d]$ for $d \in \mathbb{Z}_{\geqslant 0}$ indicates an entry of degree d. We take $\boldsymbol{\delta} = (2, 54, 95, 7)$ rather than the column degrees of $\mathbf{A}$, for reasons that we make clear below, in Section 15.2.3 and Example 15.7. Then the degree of linearization is $\delta = 40$, and we have $(\alpha_1, \alpha_2, \alpha_3, \alpha_4) = (1, 2, 3, 1)$. This implies that columns 1 and 4 of $\mathbf{A}$ will not be expanded, column 2 of $\mathbf{A}$ will be expanded into 2 columns, and column 3 of $\mathbf{A}$ will be expanded into 3 columns. Thus, $\widetilde{m} = 7$ and the indices of the columns containing the high-degree entries for each column of $\mathbf{A}$ are $(\rho_1, \rho_2, \rho_3, \rho_4) = (1, 5, 7, 4)$. Then

$$\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A}) = \begin{bmatrix} [2] & [10] & [39] & [5] & 0 & [23] & 0 \\ [75] & [39] & [39] & [69] & [11] & [39] & [15] \\ [4] & [5] & [39] & [7] & 0 & [8] & 0 \\ [10] & [39] & [39] & [6] & [14] & [35] & 0 \\ 0 & -X^{40} & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -X^{40} & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -X^{40} & 1 \end{bmatrix}. \qquad ✍$$

The key observations now are that the dimensions of $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$ are not much larger than that of $\mathbf{A}$, and that shifted Popov forms of $\mathbf{A}$ can be retrieved as the trailing submatrix of shifted Popov forms of $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$.

Lemma 15.5. *Let $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$ be nonsingular, let $\mathbf{s} \in \mathbb{Z}^m$, and let $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ and $\mathbf{H} \in \mathbb{K}[X]^{m\times m}$ be the $\mathbf{s}$-Popov form and the Hermite form of $\mathbf{A}$. Let further $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$, and let $\mathcal{E} = \begin{bmatrix} \mathbf{I} \\ \mathbf{E} \end{bmatrix}$ and $\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}} \in \mathbb{K}[X]^{(\widetilde{m}-m)\times\widetilde{m}}$ be as in Definition 15.2. Then,*

(i) $m \leqslant \widetilde{m} < 2m$;

(ii) if $\operatorname{cdeg}(\mathbf{A}) \leqslant \boldsymbol{\delta}$ componentwise then $\deg(\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})) \leqslant \delta = 1 + \lfloor |\boldsymbol{\delta}|/m \rfloor$;

(iii) for any $\mathbf{t} \in \mathbb{Z}^{\widetilde{m}-m}$, the $(\mathbf{s}, \mathbf{t})$-Popov form of $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A}) \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{E} & \mathbf{I} \end{bmatrix}$ is $\begin{bmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}$;

(iv) for any $\mathbf{t} \in \mathbb{Z}^{\widetilde{m}-m}$, if $\min(\mathbf{t}) \geqslant \max(\mathbf{s}) + \deg(\mathbf{P})$ then the $(\mathbf{s}, \mathbf{t})$-Popov form of $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$ is $\begin{bmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{R} & \mathbf{I} \end{bmatrix}$ where $\mathbf{R}$ is the remainder of $-\mathbf{E}$ modulo $\mathbf{P}$;

(v) the Hermite form of $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$ *is* $\begin{bmatrix}\mathbf{H} & \mathbf{0}\\ \mathbf{R} & \mathbf{I}\end{bmatrix}$*, where* $\mathbf{R}$ *is the remainder of* $-\mathbf{E}$ *modulo* $\mathbf{H}$*.*

Proof. Using the notation in Definition 15.2, we have $\delta = 1 + \lfloor |\boldsymbol{\delta}|/m \rfloor$, as well as $\alpha_i = 1 + \lfloor \delta_i/\delta \rfloor < 1 + m\delta_i/|\boldsymbol{\delta}|$ for all i. Hence

$$\widetilde{m} = \alpha_1 + \cdots + \alpha_m < \sum_{1 \leqslant i \leqslant m} (1 + m\delta_i/|\boldsymbol{\delta}|) = 2m$$

which gives part (i).

Part (ii) directly follows from the construction of $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$: the i-th column of $\mathbf{A}$ is split into α_i columns in $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$, all of degree at most δ except possibly one, which has degree at most $\beta_i < \delta$ if we have $\boldsymbol{\delta} \geqslant \operatorname{cdeg}(\mathbf{A})$ componentwise.

Concerning (iii), the matrix $\begin{bmatrix}\mathbf{P} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}$ is obviously in $(\mathbf{s},\mathbf{t})$-Popov form; it remains to prove that it is left-unimodularly equivalent to $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A}) \begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{E} & \mathbf{I}\end{bmatrix}$.

Let $\mathbf{T}$ denote the $(\widetilde{m}-m)\times(\widetilde{m}-m)$ submatrix of $\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}}$ formed by its rightmost columns. Since $\mathbf{T}$ is unit lower triangular, it is unimodular. Let also $\mathbf{U}$ be the unimodular matrix such that $\mathbf{UP} = \mathbf{A}$. Then, by construction of $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$ and since $\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}}\,\mathcal{E} = \mathbf{0}$, we have

$$\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A}) \begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{E} & \mathbf{I}\end{bmatrix} = \begin{bmatrix}\mathbf{A} & *\\ \mathbf{0} & \mathbf{T}\end{bmatrix} = \begin{bmatrix}\mathbf{U} & *\\ \mathbf{0} & \mathbf{T}\end{bmatrix} \begin{bmatrix}\mathbf{P} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix},$$

with $\begin{bmatrix}\mathbf{U} & *\\ \mathbf{0} & \mathbf{T}\end{bmatrix}$ being a unimodular matrix.

Now, we prove (iii). From (ii), $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$ is left-unimodularly equivalent to

$$\begin{bmatrix}\mathbf{P} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix} \begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{E} & \mathbf{I}\end{bmatrix}^{-1} = \begin{bmatrix}\mathbf{P} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix} \begin{bmatrix}\mathbf{I} & \mathbf{0}\\ -\mathbf{E} & \mathbf{I}\end{bmatrix} = \begin{bmatrix}\mathbf{P} & \mathbf{0}\\ -\mathbf{E} & \mathbf{I}\end{bmatrix}.$$

Then, let $\mathbf{R}$ be the remainder of $-\mathbf{E}$ modulo $\mathbf{P}$, that is, the unique matrix in $\mathbb{K}[X]^{(\widetilde{m}-m)\times m}$ which has column degree bounded by the column degree of $\mathbf{P}$ and such that $-\mathbf{E} = \mathbf{QP}+\mathbf{R}$ for some matrix $\mathbf{Q}$ (see Lemma 1.24). Then,

$$\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ -\mathbf{Q} & \mathbf{I}\end{bmatrix} \begin{bmatrix}\mathbf{P} & \mathbf{0}\\ -\mathbf{E} & \mathbf{I}\end{bmatrix} = \begin{bmatrix}\mathbf{P} & \mathbf{0}\\ \mathbf{R} & \mathbf{I}\end{bmatrix}$$

is left-unimodularly equivalent to $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$. Now, since $\deg(\mathbf{R}) < \deg(\mathbf{P})$, the right-hand side is in $(\mathbf{s},\mathbf{t})$-ordered weak Popov form by choice of $\mathbf{t}$; since furthermore $\operatorname{cdeg}(\mathbf{R}) < \operatorname{cdeg}(\mathbf{P})$, this right-hand side is in $(\mathbf{s},\mathbf{t})$-Popov form.

The item (v) follows from (iv) since the Hermite form of $\mathbf{A}$ is its $\mathbf{h}$-Popov form for the shift $\mathbf{h} = (0, m\deg(\mathbf{A}), \ldots, m(m-1)\deg(\mathbf{A}))$. Then, (iv) shows that for a well-chosen shift $\mathbf{t}$, the matrix $\begin{bmatrix}\mathbf{H} & \mathbf{0}\\ \mathbf{R} & \mathbf{I}\end{bmatrix}$ is in $(\mathbf{h},\mathbf{t})$-Popov form and left-unimodularly equivalent to $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$; since it is in Hermite form, the conclusion follows. ☙

To use the item (iv) in the usual case where $\deg(\mathbf{P})$ is unknown, one may choose the shift $\mathbf{t}$ by relying on the inequalities $\deg(\mathbf{P}) \leqslant \deg(\det(\mathbf{P})) = \deg(\det(\mathbf{A})) \leqslant m\deg(\mathbf{A})$.

15.2.2 Row partial linearization

The previous section shows how to reduce the non-uniformity of the degrees of the columns of $\mathbf{A}$, at least if one takes the linearization parameters as $\boldsymbol{\delta} = \operatorname{cdeg}(\mathbf{A})$. Now, we perform a similar action on the rows of $\mathbf{A}$.

For a matrix $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$ and a tuple $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$, one defines the *row partial linearization* $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ of $\mathbf{A}$ in a similar way as its column partial linearization. Precisely, these linearizations are linked by the identity $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A}) = \mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A}^{\mathsf{T}})^{\mathsf{T}}$. However, we cannot directly rely on the results above as we are interested in *left*-unimodular equivalence (that is, using row operations) and not *right*-unimodular equivalence (that is, using column operations) we cannot simply make use of the results from the previous section. Instead here we give properties of the *row* partial linearization of $\mathbf{A}$ related to *left*-unimodular equivalence.

Lemma 15.6. *Let $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$ be nonsingular, let $\mathbf{s} \in \mathbb{Z}^m$, and let $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ and $\mathbf{H} \in \mathbb{K}[X]^{m\times m}$ be the $\mathbf{s}$-Popov form and the Hermite form of $\mathbf{A}$. Let further $\boldsymbol{\delta} \in \mathbb{Z}^m_{\geqslant 0}$. Then,*

(i) for any shift $\mathbf{t} \in \mathbb{Z}^{\widetilde{m}-m}$ such that $\min(\mathbf{t}) \geqslant \max(\mathbf{s}) + \deg(\mathbf{P})$, the $(\mathbf{s},\mathbf{t})$-Popov form of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ has the form $\begin{bmatrix} \mathbf{P} & \mathbf{0} \\ * & \mathbf{I} \end{bmatrix}$;

(ii) the Hermite form of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ has the form $\begin{bmatrix} \mathbf{H} & \mathbf{0} \\ * & \mathbf{I} \end{bmatrix}$.

Proof. Let $\mathcal{E} = \begin{bmatrix} \mathbf{I} \\ \mathbf{E} \end{bmatrix}$ and $\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}} \in \mathbb{K}[X]^{(\widetilde{m}-m)\times\widetilde{m}}$ be as in Definition 15.2. Let then $\mathbf{T}$ denote the transpose of the $(\widetilde{m}-m)\times(\widetilde{m}-m)$ submatrix of $\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}}$ formed by its rightmost columns. Since $\mathbf{T}$ is unimodular, and since $\mathcal{E}^{\mathsf{T}}\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}}^{\mathsf{T}} = \mathbf{0}$, we have by construction

$$\begin{bmatrix} \mathbf{I} & \mathbf{E}^{\mathsf{T}} \\ \mathbf{0} & \mathbf{T}^{-1} \end{bmatrix} \mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A}) = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{B} & \mathbf{I} \end{bmatrix}$$

where $\mathbf{B} \in \mathbb{K}[X]^{(\widetilde{m}-m)\times m}$ is the bottom-left block of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ left-multiplied by $\mathbf{T}^{-1}$.

Now, let $\mathbf{R} \in \mathbb{K}[X]^{(\widetilde{m}-m)\times m}$ be the remainder of $\mathbf{B}$ modulo $\mathbf{P}$ (see Lemma 1.24), let $\mathbf{Q}$ be the matrix such that $\mathbf{B} = \mathbf{QP} + \mathbf{R}$, and let $\mathbf{U}$ denote the unimodular matrix such that $\mathbf{P} = \mathbf{UA}$. Then,

$$\begin{bmatrix} \mathbf{U} & \mathbf{0} \\ -\mathbf{QU} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{B} & \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{R} & \mathbf{I} \end{bmatrix}$$

is left-unimodularly equivalent to $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$. Finally, since $\deg(\mathbf{R}) < \deg(\mathbf{P})$ and $\operatorname{cdeg}(\mathbf{R}) < \operatorname{cdeg}(\mathbf{P})$, the choice of $\mathbf{t}$ implies that $\begin{bmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{R} & \mathbf{I} \end{bmatrix}$ is in $(\mathbf{s},\mathbf{t})$-Popov form.

Concerning the case of the Hermite form, the exact same arguments as in the proof of Lemma 15.5 give the conclusion. ☙

15.2.3 Reducing the degrees in shifted Popov form computation

The properties of row and column partial linearizations given above imply that, for any tuples $\boldsymbol{\delta}$ and $\boldsymbol{\gamma}$ and for well-chosen shifts $\mathbf{t}, \mathbf{u}$, the $\mathbf{s}$-Popov form of $\mathbf{A}$ appears as a submatrix of the $(\mathbf{s}, \mathbf{t}, \mathbf{u})$-Popov form of $\hat{\mathbf{A}} = \mathcal{L}^{\mathrm{r}}_{\boldsymbol{\gamma}}(\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A}))$. Furthermore, from part (i) of Lemma 15.5 we also know that $\hat{\mathbf{A}}$ is $\hat{m} \times \hat{m}$ with $m \leqslant \hat{m} < 4m$. Thus, to obtain Theorem 3.5, it remains to give a choice of $\boldsymbol{\delta}$ and $\boldsymbol{\gamma}$ such that the degree of $\hat{\mathbf{A}}$ is at most $2(1 + D_{\mathbf{A}}/m)$. For this, we use ideas from [GSSV12, Section 6].

Let $\pi_1, \pi_2 \in \mathbb{K}^{m \times m}$ be permutation matrices such that $\pi_1 \mathbf{A} \pi_2 = [\tilde{a}_{ij}]_{i,j}$ satisfies

$$\deg(\tilde{a}_{ii}) \geqslant \deg(\tilde{a}_{i'j'}) \text{ for all } 1 \leqslant i \leqslant m \text{ and } i \leqslant i', j' \leqslant m. \tag{15.4}$$

Of course these permutations (π_1, π_2) may not be unique: in what follows, we fix any such pair of permutations. It can be found in $\mathcal{O}\tilde{}(m^2)$ integer comparisons by sorting the m^2 triples $\{(\deg(a_{ij}), i, j), 1 \leqslant i, j \leqslant m\}$ in non-increasing order, where a_{ij} is the coefficient of $\mathbf{A}$ at index (i, j).

If we define $\mathbf{d} = (d_1, \ldots, d_m) \in \mathbb{Z}^m_{\geqslant 0}$ by

$$d_i = \begin{cases} \deg(\tilde{a}_{ii}) & \text{if } \tilde{a}_{ii} \neq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{15.5}$$

then $d_1 + \cdots + d_m \leqslant D_{\mathbf{A}}$ by definition of $D_{\mathbf{A}}$ in Eq. (15.1). Set $\boldsymbol{\delta} = \mathbf{d}\pi_2^{-1}$, where $\mathbf{d}$ is seen as a row vector. Then $\boldsymbol{\gamma} = \mathrm{rdeg}(\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A}))$ leads to the desired degree property. Before proving this, let us first observe it on the matrix of Example 15.4.

Example 15.7 (Reducing the input degrees). Let $\mathbf{A}$ be the matrix from Example 15.4. One can verify that $D_{\mathbf{A}} = 199 = 75 + 54 + 63 + 7$. Also

$$\begin{aligned} \pi_1 \mathbf{A} \pi_2 &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} [2] & [10] & [63] & [5] \\ [75] & [51] & [95] & [69] \\ [4] & [5] & [48] & [7] \\ [10] & [54] & [75] & [6] \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \\ &= \begin{bmatrix} [95] & [51] & [69] & [75] \\ [75] & [54] & [6] & [10] \\ [48] & [5] & [7] & [4] \\ [63] & [10] & [5] & [2] \end{bmatrix}. \end{aligned}$$

This gives $\mathbf{d} = (95, 54, 7, 2)$ and $\boldsymbol{\delta} = (2, 54, 95, 7)$, with $95 + 54 + 7 + 2 = 158 \leqslant D_{\mathbf{A}}$. The matrix $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$ has its degrees as shown in Example 15.4, and in particular we have

$$\boldsymbol{\gamma} = \mathrm{rdeg}(\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})) = (39, 75, 39, 39, 40, 40, 40).$$

Thus, we obtain

$$\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\gamma}}(\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})) = \begin{bmatrix} [2] & [10] & [39] & [5] & 0 & [23] & 0 & 0 \\ [44] & [39] & [39] & [44] & [11] & [39] & [15] & -X^{45} \\ [4] & [5] & [39] & [7] & 0 & [8] & 0 & 0 \\ [10] & [39] & [39] & [6] & [14] & [35] & 0 & 0 \\ 0 & -X^{40} & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -X^{40} & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -X^{40} & 1 & 0 \\ [30] & 0 & 0 & [24] & 0 & 0 & 0 & 1 \end{bmatrix},$$

which has degree $45 = \lceil |\boldsymbol{\gamma}|/7 \rceil$. This is less than $50 = \lceil D_{\mathbf{A}}/4 \rceil$. ✍

Coming back to the general case, we want to prove that $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\gamma}}(\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})) \in \mathbb{K}[X]^{\hat{m}\times\hat{m}}$ has degree at most $3(1 + D_{\mathbf{A}}/m)$. First, we show that the sum of the components of $\boldsymbol{\gamma}$ is less than $2m + 3D_{\mathbf{A}}$. Indeed, writing $\widetilde{m}$ for the dimension of $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$, by construction the last $\widetilde{m} - m$ rows of $\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$ have degree at most $1 + |\mathbf{d}|/m$ and its first m rows have degree at most $[\max(1 + |\mathbf{d}|/m, (\pi_1^{-1}\mathbf{d})_i)]_{1\leqslant i\leqslant m}$ componentwise, where $\mathbf{d}$ is considered as a column vector and $(\pi_1^{-1}\mathbf{d})_i$ stands for the i-th entry of $\pi_1^{-1}\mathbf{d}$. Therefore,

$$|\boldsymbol{\gamma}| \leqslant \widetilde{m}(1 + |\mathbf{d}|/m) + |\pi_1^{-1}\mathbf{d}| < 2m + 3|\mathbf{d}| \leqslant 2m + 3D_{\mathbf{A}}.$$

Then, since $\boldsymbol{\gamma} = \mathrm{rdeg}(\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A}))$ the result in item (ii) of Lemma 15.5 translated for row partial linearization ensures that the degree of the matrix $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\gamma}}(\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A}))$ is at most

$$1 + \lfloor |\boldsymbol{\gamma}|/m \rfloor < 1 + \frac{2m + 3D_{\mathbf{A}}}{m} = 3(1 + D_{\mathbf{A}}/m).$$

15.3 Fast, probabilistic computation of the shifted Popov form

Now, we present our fast algorithm to compute the shifted Popov form of polynomial matrix. It combines the partial linearization above with two main tools: known algorithms related to Smith form computation, and the fast algorithm of Chapter 8 for finding shifted Popov solution bases.

We first put aside the partial linearization, and show the correctness of the following two-step approach for computing the $\mathbf{s}$-Popov form of a nonsingular $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$:

- Compute the Smith form of $\mathbf{A}$, giving the moduli $\mathfrak{M}$, and compute a corresponding right unimodular transformation, giving the equations $\mathbf{F}$;
- Return the $\mathbf{s}$-Popov solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$.

Lemma 15.8. *Let $\mathbf{A} \in \mathbb{K}[X]^{m\times m}$ be nonsingular and let $\mathbf{S} = \mathbf{UAV}$ be the Smith form of $\mathbf{A}$, where $\mathbf{U}$ and $\mathbf{V}$ are unimodular matrices. Then, let $\mathfrak{M} \in \mathbb{K}[X]^m_{\neq 0}$ and $\mathbf{F} \in \mathbb{K}[X]^{m\times m}$ be such that $\mathbf{S} = \mathrm{diag}(\mathfrak{M})$ and $\mathbf{F} = \mathbf{V} \bmod \mathfrak{M}$. Then $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ is the row space of $\mathbf{A}$.*

Proof. Equivalently, we want to prove that $\mathbf{A}$ is a solution basis of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$. Consider a polynomial vector $\mathbf{p} \in \mathbb{K}[X]^{1\times m}$. If $\mathbf{p}$ is in the row space of $\mathbf{A}$, then $\mathbf{p}$ is a solution of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ since $\mathbf{AV} = \mathbf{U}^{-1}\mathrm{diag}(\mathfrak{M})$ with $\mathbf{U}^{-1}$ over $\mathbb{K}[X]$. On the other hand, if $\mathbf{pF} = 0 \bmod \mathfrak{M}$, then $\mathbf{pV} = \mathbf{qS}$ for some $\mathbf{q}$, and therefore $\mathbf{p} = \mathbf{qUA}$, which is in the row space of $\mathbf{A}$. ☙

Concerning the cost of the first step, the Smith factors $\mathfrak{M}$ and the reduced right-unimodular transformation $\mathbf{F}$ can be obtained in expected $\tilde{\mathcal{O}}(m^\omega \deg(\mathbf{A}))$ operations [Gup11, Theorem 4.8]. To summarize, this is done by computing

a. $\mathbf{R}$ a row reduced form of $\mathbf{A}$ [GSSV12, Theorem 18],

b. $\mathrm{diag}(\mathfrak{M})$ the Smith form of $\mathbf{R}$ [Sto03, Algorithm 12],

c. $(*, \mathbf{F})$ a reduced Smith transform for $\mathbf{R}$ [Gup11, Figure 3.2];

we remark that Steps **b.** and **c.** should be performed in conjunction with the preconditioning techniques detailed in [KKS90] (see for example the proof of [Gup11, Theorem 4.8] and the algorithm in [Gup11, Figure 6.1]). Of course, one may take for $\mathfrak{M}$ only the non-trivial Smith factors, and for $\mathbf{F}$ only the nonzero columns of the reduced transform.

Since the product of the moduli in $\mathfrak{M}$ is $\det(\mathbf{A})$, up to multiplication by a nonzero constant from $\mathbb{K}$, the sum of their degrees is $\deg(\det(\mathbf{A}))$. Then, according to Theorem 2.22, the algorithm outlined above costs allow us to compute the shifted Popov form of $\mathbf{A}$ in expected $\tilde{\mathcal{O}}(m^\omega \deg(\mathbf{A}))$ field operations. Introducing back the preliminary partial linearization of $\mathbf{A}$, we now give the details of the full algorithm.

Proposition 15.9. *Algorithm 36 is correct. If the cardinality of the field $\mathbb{K}$ is at least $8(4m)^3(3+3D_{\mathbf{A}}/m)$, then this algorithm computes the $\mathbf{s}$-Popov form of $\mathbf{A}$ in a Las Vegas fashion, using an expected number of $\tilde{\mathcal{O}}(m^\omega \lceil D_{\mathbf{A}}/m \rceil)$ operations in $\mathbb{K}$.*

Proof. We use notation from the algorithm. According to Lemma 15.5, the $\mathbf{s}$-Popov form $\mathbf{P}$ of $\mathbf{A}$ is the principal $m \times m$ submatrix of the $\mathbf{u}$-Popov form of the matrix $\widetilde{\mathbf{A}}$. This result also ensures that the $\mathbf{u}$-Popov form of $\widetilde{\mathbf{A}}$ has degree at most $\deg(\mathbf{P}) \leqslant m \deg(\mathbf{A})$. As a consequence, the choice of $\mathbf{v}$ is such that, according to Lemma 15.6, the $\mathbf{u}$-Popov form of $\widetilde{\mathbf{A}}$ is the principal $\widetilde{m} \times \widetilde{m}$ submatrix of the $\mathbf{v}$-Popov form $\mathbf{Q}$ of $\hat{\mathbf{A}}$. Thus, $\mathbf{P}$ is the principal $m \times m$ submatrix of $\mathbf{Q}$.

On the other hand, Lemma 15.8 implies that the $\mathbf{v}$-Popov solution basis $\hat{\mathbf{P}}$ of $\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$ computed at Step **3** is also the $\mathbf{v}$-Popov form of $\hat{\mathbf{A}}$, that is, $\hat{\mathbf{P}} = \mathbf{Q}$, hence the correctness.

Let us now prove the cost bound. Steps **1** and **4** do not use field operations. From the discussion in Section 15.2.3, we have that $\hat{\mathbf{A}}$ is as in Theorem 3.5: its degree $\deg(\hat{\mathbf{A}})$ is less than $3(1+D_{\mathbf{A}}/m)$, and its dimension $\hat{m}$ is less than $4m$. Then, the computation of $\mathbf{S}$ and $\mathbf{F}$ in Step **2** can be done in a Las Vegas fashion in an expected number of $\tilde{\mathcal{O}}(m^\omega \lceil D_{\mathbf{A}}/m \rceil)$ operations in $\mathbb{K}$ [Gup11, Theorem 4.8] (the preconditioning is detailed in the proof of that result, as well as in [Gup11, Figure 6.1]). This relies in particular on [Sto03, Algorithm 12] for Smith form computation.

On the other hand, we have

$$\deg(\mathfrak{m}_1) + \cdots + \deg(\mathfrak{m}_n) = \deg(\det(\mathbf{S})) = \deg(\det(\hat{\mathbf{A}})) = \deg(\det(\mathbf{A}));$$

Algorithm 36 – SPopovForm
(Shifted Popov form of a polynomial matrix)

Input:

- a nonsingular matrix $\mathbf{A} \in \mathbb{K}[X]^{m \times m}$,
- a shift $\mathbf{s} \in \mathbb{Z}^m$.

Output: the $\mathbf{s}$-Popov form of $\mathbf{A}$.

1. `/* Partial linearization */`
   $\pi_1, \pi_2 \in \mathbb{K}^{m \times m} \leftarrow$ permutations such that $\pi_1 \mathbf{A} \pi_2$ satisfies Eq. (15.4)
   $\mathbf{d} \in \mathbb{Z}^m_{\geqslant 0} \leftarrow$ nonnegative diagonal degrees of $\pi_1 \mathbf{A} \pi_2$ as in Eq. (15.5)
   $\widetilde{\mathbf{A}} \in \mathbb{K}[X]^{\widetilde{m} \times \widetilde{m}} \leftarrow \mathcal{L}^{\mathrm{c}}_{\mathbf{d}\pi_2^{-1}}(\mathbf{A})$
   $\mathbf{u} \in \mathbb{Z}^{\widetilde{m}} \leftarrow (\mathbf{s}, \max(\mathbf{s}) + m \deg(\mathbf{A}), \ldots, \max(\mathbf{s}) + m \deg(\mathbf{A}))$
   $\hat{\mathbf{A}} \in \mathbb{K}[X]^{\hat{m} \times \hat{m}} \leftarrow \mathcal{L}^{\mathrm{r}}_{\mathrm{rdeg}(\widetilde{\mathbf{A}})}(\widetilde{\mathbf{A}})$
   $\mathbf{v} \in \mathbb{Z}^{\hat{m}} \leftarrow (\mathbf{u}, \max(\mathbf{s}) + 2m \deg(\mathbf{A}), \ldots, \max(\mathbf{s}) + 2m \deg(\mathbf{A}))$
2. `/* Reduced Smith transform - details in [Gup11, Theorem 4.8] */`
   $\mathbf{S} \leftarrow$ the Smith form of $\mathbf{A}$
   $\mathbf{G} \leftarrow$ matrix such that $\mathbf{G} = \mathbf{V} \bmod \mathbf{S}$ for a right Smith transform $\mathbf{V}$
   $\mathfrak{M} \leftarrow$ nontrivial factors $(\mathfrak{m}_1, \ldots, \mathfrak{m}_n)$ of $\mathbf{S}$ `//` $\mathbf{S} = \mathrm{diag}(1, \ldots, 1, \mathfrak{M})$
   $\mathbf{F} \in \mathbb{K}[X]^{\hat{m} \times n} \leftarrow$ nontrivial columns of $\mathbf{G}$ `//` $\mathbf{G} = [\mathbf{0} \;\; \mathbf{F}]$
3. `/* Solution basis computation */`
   $\hat{\mathbf{P}} \leftarrow$ FastPopovSolBas$(\mathfrak{M}, \mathbf{F}, \mathbf{v})$ `// Algorithm 18`
4. Return the principal $m \times m$ submatrix of $\hat{\mathbf{P}}$

hence, according to Proposition 8.17, Step **3** uses $\tilde{\mathcal{O}}(m^{\omega-1}\deg(\det(\mathbf{A})))$ operations. Since $\deg(\det(\mathbf{A})) \leqslant D_{\mathbf{A}}$, the conclusion follows. ☙

We remark that, if one is satisfied with an algorithm with cost bound $\mathcal{O}(m^{\omega}\deg(\mathbf{A}))$, then the partial linearization at Step **1** can be skipped, and the condition on the size of the base field can be relaxed (see [Gup11, Theorem 4.8]): it is enough if its cardinality is at least $8m^3\deg(\mathbf{A})$.

16

Hermite form and determinant

The material presented in this chapter is the result of a joint work with George Labahn and Wei Zhou [LNZ16].

For a given nonsingular polynomial matrix $\mathbf{A}$ in $\mathbb{K}[X]^{m\times m}$, one can unimodularly transform $\mathbf{A}$ into a triangular form. Triangularizing a matrix is useful for solving linear systems and computing matrix operations such as determinants or normal forms. In the latter case, the best-known example is the Hermite normal form, first defined by Hermite in 1851 in the context of triangularizing integer matrices [Her51]. Here,

$$\mathbf{H} = \begin{bmatrix} h_{11} & & & \\ h_{21} & h_{22} & & \\ \vdots & \vdots & \ddots & \\ h_{m1} & \cdots & \cdots & h_{mm} \end{bmatrix}$$

with the added properties that each h_{ii} is monic and $\deg(h_{ij}) < \deg(h_{ii})$ for all $j < i$. Classical variations of triangularization include specifying row rather than column forms, in which case the unimodular matrix multiplies on the left rather than the right, and specifying upper rather than lower triangular forms.

Remark 16.1. In this chapter, unlike in many others in this document, we work with column spaces of the matrices rather than row spaces: we study the computation of Hermite forms for *right*-unimodular equivalence, thus focusing on the module generated by the columns of the input matrices. ☕

In this chapter, we give efficient algorithms to compute the Hermite form and the determinant of a nonsingular polynomial matrix. Since the Hermite form is a particular shifted Popov form for a specific shift, it can be computed efficiently by the algorithm given in Chapter 15. However this algorithm is probabilistic; our goal here is to exploit the triangular shape of the Hermite form to compute it efficiently and deterministically. From such a triangular form, it is natural to ask the question of retrieving the determinant of the matrix, which is the product of the diagonal entries up to some constant factor that remains to be determined. We show how this can be done via a modification of our triangularization algorithm.

In Section 16.1 we give preliminary information on column bases of polynomial matrices. Section 16.2 contains our fast algorithm for finding the diagonal entries of a triangular

form, followed in Section 16.3 by a modification in order to find the determinant. The computation of the whole Hermite form is then given in Section 16.4; it exploits the known diagonal degrees of the Hermite form to solve the problem via fast deterministic column reduction.

16.1 Preliminaries: column bases

Here, we present one of the building blocks used in our algorithms, namely the concept of *column basis* for a polynomial matrix.

A column basis of a matrix $\mathbf{A} \in \mathbb{K}[X]^{m\times n}$ is a basis of the column space of the matrix $\mathbf{A}$. We recall that the column space of $\mathbf{A}$ is the $\mathbb{K}[X]$-module $\{\mathbf{Ap}, \mathbf{p} \in \mathbb{K}[X]^{n\times 1}\}$, which is free of rank $\rho \leqslant \min(m, n)$. Such a basis can be represented as a full rank matrix in $\mathbb{K}[X]^{m\times\rho}$ whose columns are the basis elements. As detailed in Section 1.1.1, any column basis right-multiplied by a unimodular matrix gives another column basis.

Example 16.2. Let

$$\mathbf{A} = \begin{bmatrix} 6X+1 & 2X^3+X^2+6X+1 & 3 \\ 4X^5+5X^4+4X^2+X & 6X^5+5X^4+2X^3+4 & X^4+5X^3+6X^2+5X \end{bmatrix}$$

be a 2×3 matrix over $\mathbb{Z}_7[X]$ having column degree $\mathbf{s} = (5, 5, 4)$. Then a column basis $\mathbf{B}$, and a kernel basis $\mathbf{N}$, of $\mathbf{A}$ are given by

$$\mathbf{B} = \begin{bmatrix} 5X+5 & 1 \\ 3 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{N} = \begin{bmatrix} 6X^6+4X^5+5X^4+3X^3+4X^2+1 \\ 4X^4+5X^3+X^2+6X \\ 4X^7+4X^6+4X^5+4X^3+5X^2+3X+2 \end{bmatrix}.$$

For example, if $\mathbf{b}_1$ and $\mathbf{b}_2$ denote the columns of $\mathbf{B}$, then the third column of $\mathbf{A}$, denoted by $\mathbf{a}_3$, is given by

$$\mathbf{a}_3 = (4X^3+3X^2+6X+5)\,\mathbf{b}_1 \;+\; (X^4+4X^2+X+6)\,\mathbf{b}_2.$$

Here $\mathrm{cdeg}_{\mathbf{s}}(\mathbf{N}) = (11)$. Besides, the shifted leading matrix

$$\mathrm{lm}_{\mathbf{s}}(\mathbf{N}) = \begin{bmatrix} 6 \\ 0 \\ 4 \end{bmatrix}$$

has full rank, hence we have that $\mathbf{N}$ is an $\mathbf{s}$-minimal kernel basis of $\mathbf{A}$. ✍

Fast algorithms for kernel basis computation and column basis computation are given in [ZLS12] and in [ZL13], respectively. In both cases they make use of fast methods from [BL94, GJV03, ZL12] for computing approximant bases, and they admit the following cost bounds.

Theorem 16.3. *Let $\mathbf{A} \in \mathbb{K}[X]^{m\times n}$ with $m \leqslant n$ and $m \in \Theta(n)$, and let $\mathbf{s} = \mathrm{cdeg}(\mathbf{A})$. Then, there exist deterministic algorithms which compute*

(i) an $\mathbf{s}$-minimal kernel basis for $\mathbf{A}$ using $\tilde{\mathcal{O}}(n^\omega\lceil s\rceil)$ field operations,

(ii) a column basis of $\mathbf{A}$ *using* $\tilde{\mathcal{O}}(n^\omega \lceil s \rceil)$ *field operations,*

where $s = |\mathbf{s}|/n$ *is the average column degree of* $\mathbf{A}$.

For more details, we refer the reader to [ZLS12, Theorem 4.1] and [ZL13, Theorem 5.6]. We remark that, whereas we gave in Section 12.3 a detailed cost bound with logarithmic factors concerning kernel basis computation, we do not yet have such a detailed cost for column basis computation. Therefore, in this chapter, we will omit logarithmic factors in the costs.

16.2 Computing the diagonal entries of a triangular form

In this section we show how to determine the diagonal entries of a triangular form of a nonsingular matrix $\mathbf{A} \in \mathbb{K}[X]^{n\times n}$ with $\mathbf{A}$ having column degree $\mathbf{s}$. Our algorithm makes use of fast kernel and column bases computations.

As mentioned in Section 3.2.3, we consider unimodularly transforming $\mathbf{A}$ to

$$\mathbf{AU} = \mathbf{B} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ * & \mathbf{B}_2 \end{bmatrix} \tag{16.1}$$

which eliminates a top right block and gives two square diagonal blocks $\mathbf{B}_1$ and $\mathbf{B}_2$ in $\mathbf{B}$. After this block triangularization step, the matrix is now closer to being in triangular form. Applying this procedure recursively to $\mathbf{B}_1$ and $\mathbf{B}_2$, until the matrices reach dimension 1, gives the diagonal entries of a triangular form of $\mathbf{A}$. These entries are unique up to multiplication by a nonzero constant from $\mathbb{K}$, and in particular making them monic yields the diagonal entries of the Hermite form of $\mathbf{A}$.

In this procedure, a major problem is that the degrees in the unimodular multiplier $\mathbf{U}$ can be too large for efficient computation. For example, the matrix

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ -X^d & 1 & 0 & \cdots & 0 \\ 0 & -X^d & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -X^d & 1 \end{bmatrix} \in \mathbb{K}[X]^{n\times n}$$

of degree $d > 0$ is unimodular and hence its Hermite form is the identity. However the corresponding unimodular multiplier is

$$\mathbf{U} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ X^d & 1 & 0 & \cdots & 0 \\ X^{2d} & X^d & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ X^{(n-1)d} & \cdots & X^{2d} & X^d & 1 \end{bmatrix},$$

with the sum of the degrees in $\mathbf{U}$ being in $\Theta(n^3 d)$, beyond our target cost $\mathcal{O}(n^\omega d)$.

16.2.1 Fast block elimination

Our approach is to use fast kernel and column basis methods to efficiently compute the diagonal blocks $\mathbf{B}_1$ and $\mathbf{B}_2$ while at the same time avoiding the computation of all of $\mathbf{U}$.

Partition $\mathbf{A} = \begin{bmatrix} \mathbf{A}_u \\ \mathbf{A}_d \end{bmatrix}$, with $\mathbf{A}_u$ and $\mathbf{A}_d$ consisting of the upper $\lceil n/2 \rceil$ and lower $\lfloor n/2 \rfloor$ rows of $\mathbf{A}$, respectively. Then both upper and lower parts have full rank since $\mathbf{A}$ is assumed to be nonsingular. By partitioning $\mathbf{U} = \begin{bmatrix} \mathbf{U}_\ell & \mathbf{U}_r \end{bmatrix}$, where the column dimension of $\mathbf{U}_\ell$ matches the row dimension of $\mathbf{A}_u$, then $\mathbf{AU} = \mathbf{B}$ becomes

$$\begin{bmatrix} \mathbf{A}_u \\ \mathbf{A}_d \end{bmatrix} \begin{bmatrix} \mathbf{U}_\ell & \mathbf{U}_r \end{bmatrix} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ * & \mathbf{B}_2 \end{bmatrix}.$$

Notice that the matrix $\mathbf{B}_1$ is nonsingular and is therefore a column basis of $\mathbf{A}_u$. As such this can be efficiently computed as mentioned in Theorem 16.3. Then, in order to compute $\mathbf{B}_2 = \mathbf{A}_d\mathbf{U}_r$, notice that the matrix $\mathbf{U}_r$ is a right kernel basis for $\mathbf{A}_u$, which makes the top right block of $\mathbf{B}$ zero.

The following lemma states that the kernel basis $\mathbf{U}_r$ can be replaced by any other kernel basis for $\mathbf{A}_u$ thus giving another unimodular matrix that also works.

Lemma 16.4. *Partition* $\mathbf{A} = \begin{bmatrix} \mathbf{A}_u \\ \mathbf{A}_d \end{bmatrix}$ *and suppose* $\mathbf{B}_1$ *is a column basis of* $\mathbf{A}_u$ *and* $\mathbf{N}$ *a kernel basis for* $\mathbf{A}_u$*. Then there is a unimodular matrix* $\mathbf{U} = \begin{bmatrix} * & \mathbf{N} \end{bmatrix}$ *such that*

$$\mathbf{AU} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ * & \mathbf{B}_2 \end{bmatrix},$$

where $\mathbf{B}_2 = \mathbf{A}_d\mathbf{N}$*. If* $\mathbf{A}$ *is square and nonsingular, then* $\mathbf{B}_1$ *and* $\mathbf{B}_2$ *are also square and nonsingular.*

Proof. This follows from [ZL13, Lemma 3.1]. ☛

Note that we do not compute the blocks represented by the symbol $*$. Thus Lemma 16.4 allows us to determine $\mathbf{B}_1$ and $\mathbf{B}_2$ independently without considering the computation of the unimodular matrix. This procedure for computing the diagonal entries is presented in Algorithm 37. The cost of this algorithm is given below in Proposition 16.6.

16.2.2 Computational cost and example

Before giving a cost bound for our algorithm, let us observe its correctness on an example.

Example 16.5. Let us consider the matrix

$$\mathbf{A} = \begin{bmatrix} 6X+1 & 2X^3+X^2+6X+1 & 3 \\ 4X^5+5X^4+4X^2+X & 6X^5+5X^4+2X^3+4 & X^4+5X^3+6X^2+5X \\ 2 & 2X^5+5X^4+5X^3+6X^2 & 6 \end{bmatrix},$$

Algorithm 37 – HERMITEDIAG
(Diagonal entries of the Hermite form)
Input: a nonsingular matrix $\mathbf{A} \in \mathbb{K}[X]^{n \times n}$.
Output: the list $\mathbf{d} \in \mathbb{K}[X]^n$ of diagonal entries of the Hermite form of $\mathbf{A}$.

1. If $n = 1$:

- **a.** write $\mathbf{A} = \lambda \mathbf{d}$ with $\lambda \in \mathbb{K}$ and $\mathbf{d} \in \mathbb{K}[X]$ monic
- **b.** Return $\mathbf{d}$

2. Else:

- **a.** write $\mathbf{A} = \begin{bmatrix} \mathbf{A}_u \\ \mathbf{A}_d \end{bmatrix}$, where $\mathbf{A}_u$ consists of the top $\lceil n/2 \rceil$ rows of $\mathbf{A}$
- **b.** $\mathbf{B}_1 \leftarrow \text{COLUMNBASIS}(\mathbf{A}_u)$ `// [ZL13, Algorithm 2]`
- **c.** $\mathbf{N} \leftarrow \text{MINKERBAS}(\mathbf{A}_u, \text{cdeg}(\mathbf{A}))$ `// using Algorithm 26`
- **d.** $\mathbf{B}_2 \leftarrow \mathbf{A}_d \mathbf{N}$ `// using Algorithm 25`
- **e.** $\mathbf{d}_1 \leftarrow \text{HERMITEDIAG}(\mathbf{B}_1)$
- **f.** $\mathbf{d}_2 \leftarrow \text{HERMITEDIAG}(\mathbf{B}_2)$
- **g.** Return $[\mathbf{d}_1, \mathbf{d}_2]$

working over $\mathbb{Z}_7[X]$. Considering the matrix $\mathbf{A}_u$ formed by the top two rows of $\mathbf{A}$, then a column basis $\mathbf{B}_1$ and kernel basis $\mathbf{N}$ of $\mathbf{A}_u$ were given in Example 16.2. If $\mathbf{A}_d$ denotes the bottom row of $\mathbf{A}$, then this gives diagonal blocks

$$\mathbf{B}_1 = \begin{bmatrix} 5X+5 & 1 \\ 3 & 1 \end{bmatrix}$$

and

$$\mathbf{B}_2 = \mathbf{A}_d\mathbf{N} = \begin{bmatrix} X^9 + 2X^8 + X^7 + 4X^6 + 6X^5 + 4X^4 + 3X^3 + 3X^2 + 4X \end{bmatrix}.$$

Recursively computing with $\mathbf{B}_1$, we obtain a column basis and kernel basis of the top row $\mathbf{B}_{1,u}$ of $\mathbf{B}_1$, as

$$\tilde{\mathbf{B}}_1 = \begin{bmatrix} 1 \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{N}} = \begin{bmatrix} 1 \\ 2X+2 \end{bmatrix}.$$

If $\mathbf{B}_{1,d}$ denote the bottom row of $\mathbf{B}_1$, we get $\tilde{\mathbf{B}}_2 = \mathbf{B}_{1,d}\tilde{\mathbf{N}} = \begin{bmatrix} 2X+5 \end{bmatrix}$, which gives the second diagonal block from $\mathbf{B}_1$. Thus we have the diagonal entries of a triangular form of $\mathbf{B}_1$. On the other hand, since $\mathbf{B}_2$ is already a 1×1 matrix we do not need to do any extra work. As a result we have that $\mathbf{A}$ is unimodularly equivalent to

$$\begin{bmatrix} 1 & & \\ * & 2X+5 & \\ * & * & X^9 + 2X^8 + X^7 + 4X^6 + 6X^5 + 4X^4 + 3X^3 + 3X^2 + 4X \end{bmatrix},$$

giving, up to making them monic, the diagonal entries of the Hermite form of $\mathbf{A}$. ✍

Proposition 16.6. *Algorithm 37 is correct and uses $\tilde{\mathcal{O}}(n^\omega \lceil s \rceil)$ operations in $\mathbb{K}$, where $s = |\mathrm{cdeg}(\mathbf{A})|/n$ is the average column degree of $\mathbf{A}$.*

Proof. The correctness follows from the material in Section 16.2.1.

Concerning the cost bound, the three main operations are computing a column basis of $\mathbf{A}_u$, computing a kernel basis $\mathbf{N}$ of $\mathbf{A}_u$, and multiplying the matrices $\mathbf{A}_d\mathbf{N}$. We recall that $\mathbf{s} = \mathrm{cdeg}(\mathbf{A})$, and we set $\xi = |\mathbf{s}|$, an integer used to measure size for our problem.

For the column basis computation, by Theorem 16.3 we know that a column basis $\mathbf{B}_1$ of $\mathbf{A}_u$ can be computed in $\tilde{\mathcal{O}}(n^\omega \lceil s \rceil)$ operations. By [ZL13, Lemma 5.1] the column degrees of the column basis $\mathbf{B}_1$ are also bounded by the original column degrees $\mathbf{s}$. Similarly, from Theorem 16.3, computing a $\mathbf{s}$-minimal kernel basis $\mathbf{N}$ of $\mathbf{A}_u$ costs $\tilde{\mathcal{O}}(n^\omega \lceil s \rceil)$ operations, and by [ZLS12, Theorem 3.4] the sum of the $\mathbf{s}$-column degrees of the output kernel basis $\mathbf{N}$ is bounded by ξ.

For the matrix multiplication $\mathbf{A}_d\mathbf{N}$, we have that the sum of the column degrees of $\mathbf{A}_d$ and the sum of the $\mathbf{s}$-column degrees of $\mathbf{N}$ are both bounded by ξ, with $\mathrm{cdeg}(\mathbf{A}_d) \leqslant \mathbf{s}$. Therefore, up to inserting zero rows in $\mathbf{A}_d$ and zero columns in $\mathbf{N}$ to make them square, Proposition 12.5 applies and the multiplication can be done with a cost of $\tilde{\mathcal{O}}(n^\omega \lceil s \rceil)$ using Algorithm 25. (To be more precise, one would run this algorithm on the transposed matrices and take the transpose of the result.)

As a result, if we let the cost of Algorithm 37 be $g(n)$ for an input of dimension n then

$$g(n) \quad \in \quad \tilde{\mathcal{O}}(n^\omega \lceil s \rceil) + g(\lceil n/2 \rceil) + g(\lfloor n/2 \rfloor).$$

As $s = \xi/n$ depends on n we use $\mathcal{O}\tilde{}(n^\omega \lceil s \rceil) = \mathcal{O}\tilde{}(n^\omega(s+1)) = \mathcal{O}\tilde{}(n^{\omega-1}\xi + n^\omega)$ with ξ not depending on n. Then we solve the recurrence relation as

$$\begin{aligned} g(n) &\in \mathcal{O}\tilde{}(n^{\omega-1}\xi + n^\omega) + g(\lceil n/2 \rceil) + g(\lfloor n/2 \rfloor) \\ &\subseteq \mathcal{O}\tilde{}(n^{\omega-1}\xi + n^\omega) + 2g(\lceil n/2 \rceil) \\ &\subseteq \mathcal{O}\tilde{}(n^{\omega-1}\xi + n^\omega) = \mathcal{O}\tilde{}(n^\omega \lceil s \rceil). \end{aligned}$$

16.3 Fast computation of the determinant of a polynomial matrix

In this section, we show how to recursively and efficiently compute the determinant of a nonsingular matrix $\mathbf{A} \in \mathbb{K}[X]^{n\times n}$ having column degree $\mathbf{s} = \operatorname{cdeg}(\mathbf{A})$. Our algorithm follows a strategy similar to the recursive block triangularization in Section 16.2, making use of fast kernel basis and column basis computation.

Indeed, after unimodularly transforming $\mathbf{A}$ to

$$\mathbf{AU} = \mathbf{B} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ * & \mathbf{B}_2 \end{bmatrix}$$

as in Eq. (16.1), the determinant of $\mathbf{A}$ can be computed as

$$\det(\mathbf{A}) = \frac{\det(\mathbf{B})}{\det(\mathbf{U})} = \frac{\det(\mathbf{B}_1)\det(\mathbf{B}_2)}{\det(\mathbf{U})}, \tag{16.2}$$

which requires us to first compute $\det(\mathbf{B}_1)$, $\det(\mathbf{B}_2)$, and $\det(\mathbf{U})$. The same procedure can then be applied to compute the determinant of $\mathbf{B}_1$ and the determinant of $\mathbf{B}_2$. However, as $\mathbf{U}$ is unimodular we will handle its determinant differently. This can be repeated recursively until the dimension becomes 1.

One major obstacle for efficiency of this approach is that we do want to compute the scalar $\det(\mathbf{U})$, and as noted in Section 16.2, the degrees of the unimodular matrix $\mathbf{U}$ can be too large for efficient computation. To sidestep this issue, we will show that $\det(\mathbf{U})$ can be computed with only partial knowledge of the matrix $\mathbf{U}$. Combining this with the method of Section 16.2 to compute the matrices $\mathbf{B}_1$ and $\mathbf{B}_2$ without computing all of $\mathbf{B}$ and $\mathbf{U}$, we obtain an efficient recursive algorithm.

Remark 16.7. In some cases, the computation of the determinant is easily done from the diagonal entries of a triangular form. Indeed, let $\mathbf{A} \in \mathbb{K}[X]^{n\times n}$ be nonsingular and assume that we have computed the diagonal entries $h_{11}, \ldots, h_{nn}$ of its Hermite form. Then, $\det(\mathbf{A}) = \lambda h_{11} \cdots h_{nn}$ for some nonzero constant $\lambda \in \mathbb{K}$. If the constant coefficient of $h_{11} \cdots h_{nn}$ is nonzero, we can retrieve λ by computing the constant coefficient of $\det(\mathbf{A})$, which is found by $\mathbb{K}$-linear algebra using $\mathcal{O}(n^\omega)$ operations since $\det(\mathbf{A})(0) = \det(\mathbf{A}(0))$. More generally, if we know $\alpha \in \mathbb{K}$ such that $h_{11}(\alpha) \cdots h_{nn}(\alpha) \neq 0$, then we can deduce $\det(\mathbf{A})$ efficiently. However, this does not lead to a fast deterministic algorithm in general since it may happen that $\det(\mathbf{A})(\alpha) = 0$ for all field elements α, or that finding α with $h_{11}(\alpha) \cdots h_{nn}(\alpha) \neq 0$ is a difficult task.

We now focus on computing the determinant of $\mathbf{U}$, or equivalently, the determinant of $\mathbf{V} = \mathbf{U}^{-1}$. The column basis computation from [ZL13] for computing the $m \times m$ diagonal block $\mathbf{B}_1$ also gives $\mathbf{U}_r$, the matrix consisting of the right $(n-m)$ columns of $\mathbf{U}$, which is a right kernel basis for $\mathbf{A}_u$. In fact, this column basis computation also gives a right factor multiplied with the column basis $\mathbf{B}_1$ to give $\mathbf{A}_u$. The following lemma shows that this right factor coincides with the matrix $\mathbf{V}_u$ consisting of the top m rows of $\mathbf{V}$. The column basis computation therefore gives both $\mathbf{U}_r$ and $\mathbf{V}_u$ with no additional work.

Lemma 16.8. *Let m denote the dimension of $\mathbf{B}_1$. Then, a matrix $\mathbf{V}_u \in \mathbb{K}[X]^{m\times n}$ satisfies $\mathbf{B}_1\mathbf{V}_u = \mathbf{A}_u$ if and only if $\mathbf{V}_u$ is the submatrix of $\mathbf{V} = \mathbf{U}^{-1}$ formed by its top m rows.*

Proof. The proof follows directly from

$$\mathbf{BV} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ * & \mathbf{B}_2 \end{bmatrix} \begin{bmatrix} \mathbf{V}_u \\ \mathbf{V}_d \end{bmatrix} = \begin{bmatrix} \mathbf{A}_u \\ \mathbf{A}_d \end{bmatrix} = \mathbf{A}$$

.

While the determinant of $\mathbf{V}$ or the determinant of $\mathbf{U}$ is needed to compute the determinant of $\mathbf{A}$, a major problem is that we do not know $\mathbf{U}_\ell$ or $\mathbf{V}_d$, which may not be efficiently computed due to their possibly large degrees. This means we need to compute the determinant of $\mathbf{V}$ or $\mathbf{U}$ without knowing the complete matrix $\mathbf{V}$ or $\mathbf{U}$. The following lemma shows how this can be done using just $\mathbf{U}_r$ and $\mathbf{V}_u$, which are obtained from the computation of the column basis $\mathbf{B}_1$.

Lemma 16.9. *Let $\mathbf{U} = \begin{bmatrix} \mathbf{U}_\ell & \mathbf{U}_r \end{bmatrix}$ and $\mathbf{A}$ satisfy, as before,*

$$\mathbf{AU} = \begin{bmatrix} \mathbf{A}_u \\ \mathbf{A}_d \end{bmatrix} \begin{bmatrix} \mathbf{U}_\ell & \mathbf{U}_r \end{bmatrix} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ * & \mathbf{B}_2 \end{bmatrix} = \mathbf{B},$$

where the row dimension of $\mathbf{A}_u$, the column dimension of $\mathbf{U}_\ell$, and the dimension of $\mathbf{B}_1$ are m. Let $\mathbf{V} = \begin{bmatrix} \mathbf{V}_u \\ \mathbf{V}_d \end{bmatrix}$ be the inverse of $\mathbf{U}$ with m rows in $\mathbf{V}_u$ and $\mathbf{U}_\ell^* \in \mathbb{K}[X]^{n\times m}$ be a matrix such that $\mathbf{U}^* = \begin{bmatrix} \mathbf{U}_\ell^* & \mathbf{U}_r \end{bmatrix}$ is unimodular. Then $\mathbf{V}_u\mathbf{U}_\ell^*$ is unimodular and

$$\det(\mathbf{A}) = \frac{\det(\mathbf{B})\det(\mathbf{V}_u\mathbf{U}_\ell^*)}{\det(\mathbf{U}^*)}.$$

Proof. Since $\det(\mathbf{A}) = \det(\mathbf{B})\det(\mathbf{V})$, it is enough to show that

$$\det(\mathbf{V}) = \det(\mathbf{V}_u\mathbf{U}_\ell^*)/\det(\mathbf{U}^*).$$

This follows from

$$\begin{aligned} \det(\mathbf{V})\det(\mathbf{U}^*) &= \det(\mathbf{VU}^*) \\ &= \det\left(\begin{bmatrix} \mathbf{V}_u \\ \mathbf{V}_d \end{bmatrix} \begin{bmatrix} \mathbf{U}_\ell^* & \mathbf{U}_r \end{bmatrix}\right) \\ &= \det\left(\begin{bmatrix} \mathbf{V}_u\mathbf{U}_\ell^* & \mathbf{0} \\ * & \mathbf{I} \end{bmatrix}\right) \\ &= \det(\mathbf{V}_u\mathbf{U}_\ell^*). \end{aligned}$$

In particular $\det(\mathbf{V}_u\mathbf{U}_\ell^*)$ is a nonzero constant and thus $\mathbf{V}_u\mathbf{U}_\ell^*$ is unimodular.

Lemma 16.9 shows that the determinant of $\mathbf{V}$ can be computed using $\mathbf{V}_u$, $\mathbf{U}_r$, and a unimodular completion $\mathbf{U}^*$ of $\mathbf{U}_r$. In fact, this can be made still more efficient by noticing that since we are looking for a constant determinant, the higher degree parts of the matrices do not affect the computation.

Lemma 16.10. *If $\mathbf{U} \in \mathbb{K}[X]^{n\times n}$ is unimodular, then*

$$\det(\mathbf{U}) = \det(\mathbf{U} \bmod X) = \det(\mathbf{U}(0)).$$

Proof. Note that $\det(\mathbf{U}(\alpha)) = \det(\mathbf{U})(\alpha)$ for any $\alpha \in \mathbb{K}$, that is, the result is the same whether we do evaluation before or after computing the determinant. Then, taking $\alpha = 0$ concludes the proof. ☙

Lemma 16.10 allows us to use just the degree zero coefficient matrices in the computation. Hence Lemma 16.9 can be improved as follows.

Lemma 16.11. *Let $\mathbf{A}$, $\mathbf{U} = \begin{bmatrix}\mathbf{U}_\ell & \mathbf{U}_r\end{bmatrix}$, and $\mathbf{V} = \begin{bmatrix}\mathbf{V}_u \\ \mathbf{V}_d\end{bmatrix}$ be as before. Let $U_r = \mathbf{U}_r \bmod X$ and $V_u = \mathbf{V}_u \bmod X$ be the constant matrices of $\mathbf{U}_r$ and $\mathbf{V}_u$, respectively. Let $U_\ell^* \in \mathbb{K}^{n\times m}$ be a matrix such that $U^* = \begin{bmatrix}U_\ell^* & U_r\end{bmatrix}$ is nonsingular. Then*

$$\det(\mathbf{A}) = \frac{\det(\mathbf{B})\det(V_u U_\ell^*)}{\det(U^*)}.$$

Proof. Suppose we have a matrix $\mathbf{U}_\ell^* \in \mathbb{K}[X]^{n\times m}$ such that $U_\ell^* = \mathbf{U}_\ell^* \bmod X$ and $\mathbf{U}^* = \begin{bmatrix}\mathbf{U}_\ell^* & \mathbf{U}_r\end{bmatrix}$ is unimodular. Using Lemma 16.9 and Lemma 16.10, we have that $\mathbf{V}_u\mathbf{U}_\ell^*$ is unimodular with $V_u U_\ell^* = \mathbf{V}_u\mathbf{U}_\ell^* \bmod X$ and thus

$$\det(\mathbf{A}) = \det(\mathbf{B})\det(\mathbf{V}_u\mathbf{U}_\ell^*)/\det(\mathbf{U}^*) = \det(\mathbf{B})\det(V_u U_\ell^*)/\det(U^*).$$

Let us now show how to construct such a matrix $\mathbf{U}_\ell^*$. Let $\mathbf{W}_\ell^* \in \mathbb{K}[X]^{n\times m}$ be any matrix such that $\mathbf{W}^* = \begin{bmatrix}\mathbf{W}_\ell^* & \mathbf{U}_r\end{bmatrix}$ is unimodular and let W_ℓ^* denote its constant term $W_\ell^* = \mathbf{W}_\ell^* \bmod X$. It is easily verified that $\begin{bmatrix}W_\ell^* & U_r\end{bmatrix}^{-1}\begin{bmatrix}U_\ell^* & U_r\end{bmatrix} = \begin{bmatrix}T_u & 0 \\ T_d & I\end{bmatrix}$ for some nonsingular $T_u \in \mathbb{K}^{m\times m}$ and some $T_d \in \mathbb{K}^{n-m\times m}$. Then define the matrix $\mathbf{U}_\ell^* = \mathbf{W}_\ell^*\begin{bmatrix}T_u \\ T_d\end{bmatrix}$ in $\mathbb{K}[X]^{n\times m}$. On the one hand, we have that $\mathbf{U}^* = \begin{bmatrix}\mathbf{U}_\ell^* & \mathbf{U}_r\end{bmatrix} = \mathbf{W}^*\begin{bmatrix}T_u & 0 \\ T_d & I\end{bmatrix}$ is unimodular. On the other hand, by construction we have that $\mathbf{U}_\ell^* \bmod X = W_\ell^*\begin{bmatrix}T_u \\ T_d\end{bmatrix} = U_\ell^*$. ☙

Thus Lemma 16.11 requires us to compute $U_\ell^* \in \mathbb{K}^{n\times m}$ a matrix such that $U^* = \begin{bmatrix}U_\ell^* & U_r\end{bmatrix}$ is nonsingular. This can be obtained from the nonsingular matrix that transforms V_u to its reduced column echelon form computed using the Gauss Jordan transform algorithm from [Sto00] with a cost of $\mathcal{O}(nm^{\omega-1})$ field operations.

We now have all the ingredients needed for computing the determinant of $\mathbf{A}$. A recursive algorithm is given in Algorithm 38, which computes the determinant of $\mathbf{A}$ as the product of the determinant of $\mathbf{V}$ and the determinant of $\mathbf{B}$. The determinant of $\mathbf{B}$ is computed by recursively computing the determinants of its diagonal blocks $\mathbf{B}_1$ and $\mathbf{B}_2$.

Algorithm 38 – DETERMINANT
(Determinant of a nonsingular polynomial matrix)
Input: a nonsingular matrix $\mathbf{A} \in \mathbb{K}[X]^{n \times n}$.
Output: the determinant of $\mathbf{A}$.

1. If $n = 1$ then return $\mathbf{A}$
2. Else:
   - **a.** write $\mathbf{A} = \begin{bmatrix} \mathbf{A}_u \\ \mathbf{A}_d \end{bmatrix}$, with $\mathbf{A}_u$ consisting of the top $\lceil n/2 \rceil$ rows of $\mathbf{A}$
   - **b.** $\mathbf{B}_1, \mathbf{U}_r, \mathbf{V}_u \leftarrow$ COLUMNBASIS$(\mathbf{A}_u)$ `// [ZL13, Algorithm 2]`
     `/* Here` COLUMNBASIS `also returns the kernel basis` $\mathbf{U}_r$
     `and the right factor` $\mathbf{V}_u$ `such that` $\mathbf{A}_u = \mathbf{B}_1 \mathbf{V}_u$. `*/`
   - **c.** $\mathbf{B}_2 \leftarrow \mathbf{A}_d \mathbf{U}_r$
   - **d.** $U_r \leftarrow \mathbf{U}_r \bmod X$
   - **e.** $V_u \leftarrow \mathbf{V}_u \bmod X$
   - **f.** Compute $U_\ell^* \in \mathbb{K}^{n \times \lceil n/2 \rceil}$ such that $U^* = \begin{bmatrix} U_\ell^* & U_r \end{bmatrix}$ is nonsingular
   - **g.** $d_V \leftarrow \det(V_u U_\ell^*) / \det(U^*)$ `// element of` $\mathbb{K}$
   - **h.** $\mathbf{d_B} \leftarrow$ DETERMINANT$(\mathbf{B}_1)$DETERMINANT$(\mathbf{B}_2)$
   - **i.** Return $d_V \mathbf{d_B}$

Proposition 16.12. *Algorithm 38 is correct and uses $\tilde{\mathcal{O}}(n^\omega\lceil s\rceil)$ operations in $\mathbb{K}$, where $s = |\mathrm{cdeg}(\mathbf{A})|/n$ is the average column degree of $\mathbf{A}$.*

Proof. The correctness follows from the material in this section.

From Lemma 16.4 and Proposition 16.6 the computation of the two diagonal blocks $\mathbf{B}_1$ and $\mathbf{B}_2$ costs $\tilde{\mathcal{O}}(n^\omega\lceil s\rceil)$ operations. As mentioned above, computing U_l^* at Step **6** of the algorithm uses $\mathcal{O}(n^\omega)$ operations. Step **7** involves only constant matrices so that d_V can be computed in $\mathcal{O}(n^\omega)$. Finally, $\det(\mathbf{B}_1)$ and $\det(\mathbf{B}_2)$ are computed recursively and multiplied. Since these are two univariate polynomials of degree at most $\deg(\det(\mathbf{A})) \leqslant \xi = ns$, their product $\mathbf{d_B}$ is obtained in $\tilde{\mathcal{O}}(\xi) \subset \tilde{\mathcal{O}}(n^\omega\lceil s\rceil)$ operations.

Therefore, the recurrence relation for the cost of the Algorithm 38 is the same as that in the proof of Proposition 16.6, and the total cost is $\tilde{\mathcal{O}}(n^\omega\lceil s\rceil)$. ∎

For an input matrix $\mathbf{A}$ which has non-uniform degrees, Proposition 16.12 can be further improved using partial linearization techniques, and in particular with the following result from [GSSV12, Corollary 3] .

Lemma 16.13. *Let $\mathbf{A} \in \mathbb{K}[X]^{n\times n}$ be nonsingular. Using no operation in $\mathbb{K}$, one can build a matrix $\hat{\mathbf{A}} \in \mathbb{K}[X]^{\hat{n}\times\hat{n}}$ such that*

(i) $n \leqslant \hat{n} < 3n$ and $\deg(\hat{\mathbf{A}}) \leqslant \lceil D_{\mathbf{A}}/n\rceil$,

(ii) the determinant of $\mathbf{A}$ is equal to the determinant of $\hat{\mathbf{A}}$.

Combining this with our algorithm, this proves the result announced in Theorem 3.8.

Example 16.14. In order to observe the correctness of the algorithm, let

$$\mathbf{A} = \begin{bmatrix} -X+2 & -2X-3 & 3X^3+X^2 & -X+2 & -3X^5-X^4 \\ -X & -2 & 3X^3 & -X & -3X^5 \\ -2 & X+3 & 2 & -2 & -2X^2 \\ 0 & 1 & -3X^2-2 & -2X^2-1 & X^4+X^2 \\ 0 & 2 & 3 & -3X^2 & -2X^4-3X^2+3 \end{bmatrix}$$

working over $\mathbb{Z}_7[X]$. If $\mathbf{A}_u$ denotes the top three rows of $\mathbf{A}$, then we have a column basis

$$\mathbf{B}_1 = \begin{bmatrix} -X+2 & -2X-3 & 3X^3+X^2 \\ -X & -2 & 3X^3 \\ -2 & X+3 & 2 \end{bmatrix}$$

and a minimal kernel basis

$$\mathbf{U}_r = \begin{bmatrix} 3 & 0 \\ 0 & 0 \\ 0 & X^2 \\ -3 & 0 \\ 0 & 1 \end{bmatrix}$$

for $\mathbf{A}_u$. The second block diagonal is then given by

$$\mathbf{A}_d\mathbf{U}_r = \begin{bmatrix} X^2-3 & -2X^4-X^2 \\ -2X^2 & -2X^4+3 \end{bmatrix}.$$

The computation of the column basis $\mathbf{B}_1$ also gives the right factor

$$\mathbf{V}_u = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -X^2 \end{bmatrix}$$

and so the constant term matrices are then

$$U_r = \begin{bmatrix} 3 & 0 \\ 0 & 0 \\ 0 & 0 \\ -3 & 0 \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad V_u = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

with Gaussian-Jordan elimination used to find a nonsingular completion of U_r as

$$U_\ell^* = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

The determinant of $\mathbf{U}$ is then computed as

$$d_V = \frac{\det(V_u U_\ell^*)}{\det(U^*)} = -\frac{1}{3} = 2.$$

The determinants of $\mathbf{B}_1$ and $\mathbf{B}_2$ are computed recursively. In the case of $\mathbf{B}_1$ a minimal kernel basis and column basis are given by

$$\mathbf{U}_{r,1} = \begin{bmatrix} 3X^2 \\ 0 \\ 1 \end{bmatrix}, \quad \mathbf{B}_{1,1} = \begin{bmatrix} -X+2 & 0 \\ -X & 2X-2 \end{bmatrix}, \quad \text{and } \mathbf{V}_{u,1} = \begin{bmatrix} 1 & 2 & -3X^2 \\ 0 & 1 & 0 \end{bmatrix}.$$

This gives the remaining diagonal block as $\mathbf{B}_{1,2} = \begin{bmatrix} X^2+2 \end{bmatrix}$. The corresponding constant term matrices $U_{r,1}$ and $V_{u,1}$ and nonsingular completion $U_{\ell,1}^*$ are then given by

$$U_{r,1} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad V_{u,1} = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad \text{and } U_{\ell,1}^* = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix},$$

which gives $d_{V_1} = 1$. Hence $\det(\mathbf{B}_1) = (-X+2)(2X-2)(X^2+2)$. A similar argument gives $\det(\mathbf{B}_2) = (X^2-3)(X^4+3)$ and hence

$$\begin{aligned} \det(\mathbf{A}) &= d_V \det(\mathbf{B}_1)\det(\mathbf{B}_2) \\ &= 3X^{10} - 2X^9 + 3X^8 + 2X^7 - X^6 - X^5 + X^4 - X^3 - 2X^2 + X - 3. \end{aligned}$$

✍

16.4 Fast Hermite form algorithm with known minimal degree

In Section 16.2, we have shown how to efficiently determine the diagonal entries of the Hermite form $\mathbf{H}$ of a nonsingular input matrix $\mathbf{A} \in \mathbb{K}[X]^{n\times n}$. One then still needs to compute the remaining entries of $\mathbf{H}$.

We start this section by showing how knowing the diagonal degrees helps to obtain $\mathbf{H}$ via shifted column reduction for a well-chosen shift. This follows ideas in Section 1.2.1, noting that the diagonal degrees give in particular the diagonal entries of $\mathbf{H}$, which correspond to the shifted minimal degree of the module for the Hermite shift. However, this leads to a Hermite form algorithm that is not yet satisfactory in terms of efficiency. Namely, when the degrees of the diagonal entries have a large amplitude, the shifted column reduction step may cost $\mathcal{O}\tilde{}(n^{\omega+1}\deg(\mathbf{A}))$ field operations.

Then, knowing the diagonal degrees of $\mathbf{H}$ allows us to rely on partial linearization techniques similar to that in Section 6.2; they allow us to reduce to the case of computing a column reduced form of $\mathbf{A}$ for an almost uniform shift. Along with the algorithm in Section 16.2, this gives an algorithm to compute the Hermite form of $\mathbf{A}$ in $\mathcal{O}\tilde{}(n^{\omega}\deg(\mathbf{A}))$ field operations using fast deterministic column reduction [GSSV12].

16.4.1 Hermite form via shifted column reduction

It is known that the Hermite form $\mathbf{H}$ of $\mathbf{A}$ is a *shifted* reduced form of $\mathbf{A}$ for a whole range of shifts. Without further information on the degrees in $\mathbf{H}$, one appropriate shift is

$$\mathbf{h} \;=\; (n(n-1)d,\;\; n(n-2)d,\;\; \ldots,\;\; nd,\;\; 0) \tag{16.3}$$

where $d = \deg(\mathbf{A})$ (see [BLV06, Lemma 2.6]). We note that this shift has a large amplitude, namely $\max(\mathbf{h}) - \min(\mathbf{h}) \in \Theta(n^2 d)$. Unfortunately, there does not appear to be a deterministic shifted reduction algorithm that would compute an $\mathbf{h}$-reduced form of $\mathbf{A}$ in $\mathcal{O}\tilde{}(n^{\omega}d)$ field operations.

Now, let us consider the degrees $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_n)$ of the diagonal entries of $\mathbf{H}$. Then we have that $\mathbf{H}$ is a $-\boldsymbol{\delta}$-column reduced form of $\mathbf{A}$ and, in addition, that $\mathbf{H}$ can be easily recovered from any $-\boldsymbol{\delta}$-column reduced form of $\mathbf{A}$. More precisely, suppose that we know $\boldsymbol{\delta}$, for example thanks to the algorithm in Section 16.2. Then, we claim that $\mathbf{H}$ can be computed as follows, where $\mu = (\max(\boldsymbol{\delta}), \ldots, \max(\boldsymbol{\delta})) \in \mathbb{Z}^n_{\geqslant 0}$:

$$\mathbf{X}^{\mu-\boldsymbol{\delta}}\,\mathbf{A} \;\xrightarrow{\text{reduction}}\; \mathbf{X}^{\mu-\boldsymbol{\delta}}\,\mathbf{R} \;\xrightarrow{\text{normalization}}\; \mathbf{H} = \mathbf{R}\,\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})^{-1}$$

where $\mathbf{R}$ is any $-\boldsymbol{\delta}$-column reduced form of $\mathbf{A}$. To show this, we will rely on the following consequences of [SS11, Lemma 17].

Lemma 16.15. *Let $\mathbf{A}$ and $\mathbf{B}$ be column reduced matrices in $\mathbb{K}[X]^{n\times n}$ with uniform column degree $(d, \ldots, d)$, for some $d \in \mathbb{Z}_{\geqslant 0}$. If $\mathbf{A}$ and $\mathbf{B}$ are right-unimodularly equivalent then*

$$\mathbf{A}\,\mathrm{lm}_{\mathbf{0}}(\mathbf{A})^{-1} = \mathbf{B}\,\mathrm{lm}_{\mathbf{0}}(\mathbf{B})^{-1}.$$

Proof. The matrix $\mathbf{A}$ is column reduced with uniform column degree $(d, \dots, d)$. As such $\mathbf{A}\,\mathrm{lm}_{\mathbf{0}}(\mathbf{A})^{-1}$ is its Popov form according to [SS11, Lemma 17] (i.e. its leading coefficient matrix is the identity). Similarly, $\mathbf{B}\,\mathrm{lm}_{\mathbf{0}}(\mathbf{B})^{-1}$ is the Popov form of $\mathbf{B}$ in this case. We recall that the Popov form is a canonical form under right-unimodular equivalence for nonsingular matrices in $\mathbb{K}[X]^{n\times n}$; for a general definition we refer the reader to [Kai80]. Thus, since $\mathbf{A}$ and $\mathbf{B}$ are right-unimodularly equivalent, the uniqueness of the Popov form implies $\mathbf{A}\,\mathrm{lm}_{\mathbf{0}}(\mathbf{A})^{-1} = \mathbf{B}\,\mathrm{lm}_{\mathbf{0}}(\mathbf{B})^{-1}$. ☙

As we often wish to apply Lemma 16.15 with shifts we also include the following.

Corollary 16.16. *Let $\mathbf{s} \in \mathbb{Z}^m$ be a shift, and let $\mathbf{A}$ and $\mathbf{B}$ be $\mathbf{s}$-column reduced matrices in $\mathbb{K}[X]^{n\times n}$ with uniform $\mathbf{s}$-column degree $(d, \dots, d)$, for some $d \in \mathbb{Z}$. If $\mathbf{A}$ and $\mathbf{B}$ are right-unimodularly equivalent then*

$$\mathbf{A}\,\mathrm{lm}_{\mathbf{s}}(\mathbf{A})^{-1} = \mathbf{B}\,\mathrm{lm}_{\mathbf{s}}(\mathbf{B})^{-1}.$$

Proof. We simply replace $\mathbf{A}$ and $\mathbf{B}$ by $\mathbf{X}^{\mathbf{s}}\,\mathbf{A}$ and $\mathbf{X}^{\mathbf{s}}\,\mathbf{B}$ in the previous proof. ☙

We remark that this result is directly related to those mentioned in Section 1.2.1, and in particular Lemma 1.26. Yet, for better readability we recall this since here we are working with right unimodular equivalence.

Thus we can start with the matrix $\mathbf{X}^{\mu-\boldsymbol{\delta}}\,\mathbf{A}$, column reduce this matrix and then normalize it to get our normal form. However $\mathbf{X}^{\mu-\boldsymbol{\delta}}\,\mathbf{A}$ may have some entries of large degree. Indeed, $\max(\boldsymbol{\delta})$ may be as large as $\deg(\det(\mathbf{A}))$ while having $\min(\boldsymbol{\delta}) = 0$, in which case the degree of $\mathbf{X}^{\mu-\boldsymbol{\delta}}\,\mathbf{A}$ is at least $\deg(\det(\mathbf{A}))$. For efficient deterministic shifted column reduction we would need the degree of $\mathbf{X}^{\mu-\boldsymbol{\delta}}\,\mathbf{A}$ to be in $\mathcal{O}(\deg(\mathbf{A}))$.

16.4.2 Reducing the amplitude of the minimal degree

In the strategy presented in the previous subsection, the main obstacle to obtaining an efficient algorithm is that the diagonal degrees of $\mathbf{H}$ might have a large amplitude. In this subsection, we will show how partial linearization techniques allow us to build a matrix $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ such that $\mathbf{H}$ can be obtained from a $-\mathbf{d}$-reduced form of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ for a shift $\mathbf{d}$ that has a small amplitude.

A key fact is that the average of the degrees $\boldsymbol{\delta}$ is controlled. Namely, denoting by δ the average of $\boldsymbol{\delta}$, we have that $\delta \leqslant \deg(\mathbf{A})$. Indeed, the product of the diagonal entries of $\mathbf{H}$ is $\det(\mathbf{H})$ which, up to a constant multiplier, is the same as $\det(\mathbf{A})$ and thus the degree of this product is

$$n\delta = \delta_1 + \cdots + \delta_n = \deg(\det(\mathbf{A})) \leqslant n\deg(\mathbf{A}).$$

In order to reduce the amplitude of $\boldsymbol{\delta}$, one can split the entries that are larger than δ into several entries each at most δ. From this we obtain another tuple $\mathbf{d} = (d_1, \dots, d_{\widetilde{n}})$ with $\max(\mathbf{d}) - \min(\mathbf{d}) \leqslant \delta \leqslant \deg(\mathbf{A})$ and having length $\widetilde{n}$ less than $2n$.

Most importantly for our purpose, there is a corresponding transformation of matrices which behaves well with regards to shifted reduction. Namely, this transformation is a type of *row partial linearization*, similar to those introduced in [GSSV12, Section 6]. Let

us describe this linearization in the particular case of the Hermite form $\mathbf{H}$ of $\mathbf{A}$. For each i, we focus on the row i of $\mathbf{H}$. If its degree δ_i is larger than δ then the row is expanded into α_i rows of degree at most δ. This yields a $\widetilde{n} \times n$ matrix $\widetilde{\mathbf{H}}$ of degree at most δ. Furthermore, certain elementary columns are inserted into $\widetilde{\mathbf{H}}$ resulting in a square nonsingular matrix $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H})$ which preserves fundamental properties of $\mathbf{H}$ (for example, its Smith factors and its determinant). Namely, $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H})$ has dimension $\widetilde{n} \times \widetilde{n}$ and degree at most δ, which in this case is the average row degree of $\mathbf{H}$.

For example, suppose that $\mathbf{H}$ is a 4×4 matrix in Hermite form with diagonal entries having degrees $(2, 37, 7, 18)$. Such a matrix has degree profile

$$\mathbf{H} = \begin{bmatrix} (2) & & & \\ [36] & (37) & & \\ [6] & [6] & (7) & \\ [17] & [17] & [17] & (18) \end{bmatrix},$$

where $[d]$ stands for an entry of degree at most d and (d) stands for a monic entry of degree exactly d. Here $\mathbf{H}$ has row degree $\boldsymbol{\delta} = (2, 37, 7, 18)$.

Let us now construct the row partial linearization $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H})$. Considering the upper bound $\delta = 1 + \lfloor (2 + 37 + 7 + 18)/4 \rfloor = 17$ on the average row degree of $\mathbf{H}$, we will split the high-degree rows of $\mathbf{H}$ in several rows having degree less than δ. The first row is unchanged; the second row is expanded in two rows of degree 16 and one row of degree 3; the third row is unchanged; and finally the last row is expanded in one row of degree 16 and one row of degree 1. Then, the matrix with expanded rows is

$$\widetilde{\mathbf{H}} = \begin{bmatrix} (2) & & & \\ [16] & [16] & & \\ [16] & [16] & & \\ [2] & (3) & & \\ [6] & [6] & (7) & \\ [16] & [16] & [16] & [16] \\ [0] & [0] & [0] & (1) \end{bmatrix}.$$

Note that $\mathbf{H}$ and $\widetilde{\mathbf{H}}$ are related by the identity $\mathcal{E}\widetilde{\mathbf{H}} = \mathbf{H}$, where $\mathcal{E}$ is the *expansion-compression* matrix

$$\mathcal{E} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & X^{17} & X^{34} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & X^{17} \end{bmatrix}.$$

We can insert elementary columns in $\widetilde{\mathbf{H}}$ by

$$\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H}) = \begin{bmatrix} (2) & & & & & & \\ [16] & X^{17} & & [16] & & & \\ [16] & -1 & X^{17} & [16] & & & \\ [2] & & -1 & (3) & & & \\ [6] & & & [6] & (7) & & \\ [16] & & & [16] & [16] & X^{17} & [16] \\ [0] & & & [0] & [0] & -1 & (1) \end{bmatrix}$$

which indicate the row operations needed to keep track of the structure of the original rows of $\mathbf{H}$. Now the reduced tuple of row degrees $\mathbf{d} = (2, 17, 17, 3, 7, 17, 1)$ has as its largest entry the *average* row degree $\delta = 17$ of $\mathbf{H}$. Furthermore, $\mathbf{H}$ can be reconstructed from $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H})$, without field operations, as a submatrix of $\mathcal{E}\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H})$.

Remark 16.17. Because the purpose is different, the partial linearization used in this section is defined in a way that is slightly different from what was done in Section 15.2. It would be possible to use the same linearizations, but at the cost that the proofs would involve permutation matrices and would be considerably more difficult to follow. ☕

Formally we define the partial linearization for a matrix $\mathbf{A}$ and a tuple $\boldsymbol{\delta}$, with the latter not necessarily related to $\mathrm{rdeg}(\mathbf{A})$. Indeed, we will apply this in a situation where the tuple $\boldsymbol{\delta}$ is formed by the diagonal degrees of the Hermite form of $\mathbf{A}$.

Definition 16.18. *Let $\mathbf{A} \in \mathbb{K}[X]^{n\times n}$, $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_n) \in \mathbb{Z}^n_{\geqslant 0}$ and set*

$$\delta = 1 + \left\lfloor \frac{(\delta_1 + \cdots + \delta_n)}{n} \right\rfloor .$$

For any $i \in \{1, \ldots, n\}$ write $\delta_i = (\alpha_i - 1)\delta + \beta_i$ with $\alpha_i = \lceil \delta_i/\delta \rceil$ and $1 \leqslant \beta_i \leqslant \delta$ if $\delta_i > 0$, while $\alpha_i = 1$ and $\beta_i = 0$ if $\delta_i = 0$. Set $\widetilde{n} = \alpha_1 + \cdots + \alpha_n$ and define $\mathbf{d} \in \mathbb{Z}^{\widetilde{n}}_{\geqslant 0}$ as

$$\mathbf{d} = (\ \underbrace{\delta,\ \ldots\ ,\ \delta,\ \beta_1}_{\alpha_1},\ \ldots\ ,\ \underbrace{\delta,\ \ldots,\ \delta,\ \beta_n}_{\alpha_n}\ ) \tag{16.4}$$

as well as the row expansion-compression matrix $\mathcal{E} \in \mathbb{K}[X]^{n\times\widetilde{n}}$ as

$$\mathcal{E} = \begin{bmatrix} 1 & X^\delta & \cdots & X^{(\alpha_1-1)\delta} & & & & \\ & & & & \ddots & & & \\ & & & & & 1 & X^\delta & \cdots & X^{(\alpha_n-1)\delta} \end{bmatrix}. \tag{16.5}$$

Let $\widetilde{\mathbf{A}} \in \mathbb{K}[X]^{\widetilde{n}\times n}$ be such that $\mathbf{A} = \mathcal{E}\widetilde{\mathbf{A}}$ with all the rows of $\widetilde{\mathbf{A}}$ having degree at most δ except possibly at indices $\{\alpha_1 + \cdots + \alpha_i, 1 \leqslant i \leqslant n\}$. Define $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A}) \in \mathbb{K}[X]^{\widetilde{n}\times\widetilde{n}}$ as:

(i) for $1 \leqslant i \leqslant n$, the column $\alpha_1 + \cdots + \alpha_i$ of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ is the column i of $\widetilde{\mathbf{A}}$;

(ii) for $0 \leqslant i \leqslant n-1$ and $1 \leqslant j \leqslant \alpha_{i+1} - 1$, the column $\alpha_1 + \cdots + \alpha_i + j$ of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ is the column

$$\begin{bmatrix} 0 & \cdots & 0 & X^\delta & -1 & 0 & \cdots & 0 \end{bmatrix}^{\mathsf{T}} \in \mathbb{K}[X]^{\widetilde{n}\times 1}$$

with the entry X^δ at row index $\alpha_1 + \cdots + \alpha_i + j$.

It follows from this construction that any matrix $\mathbf{A} \in \mathbb{K}[X]^{n\times n}$ is the submatrix of $\mathcal{E}\,\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ formed by its columns at indices $\{\alpha_1 + \cdots + \alpha_i, 1 \leqslant i \leqslant n\}$.

It is important to note that this transformation has good properties regarding the computation of $-\boldsymbol{\delta}$-shifted reduced forms of $\mathbf{A}$, where $\boldsymbol{\delta}$ is the tuple of diagonal degrees of the Hermite form of $\mathbf{A}$. Indeed, it transforms any $-\boldsymbol{\delta}$-reduced form $\mathbf{R}$ of $\mathbf{A}$ into a

$-\mathbf{d}$-reduced form $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R})$ of the transformed $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$. In other words, we have the following diagram:

$$\begin{array}{ccc}
\mathbf{X}^{\mu-\boldsymbol{\delta}}\,\mathbf{A} & \xrightarrow{\text{reduction}} & -\boldsymbol{\delta}\text{-reduced form of } \mathbf{A} \\
\Big| & & \Big| \\
\text{partial linearization} & & \text{partial linearization} \\
\downarrow & & \downarrow \\
\mathbf{X}^{\mathbf{m}-\mathbf{d}}\,\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A}) & \xrightarrow{\text{reduction}} & -\mathbf{d}\text{-reduced form of } \mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})
\end{array},$$

where $\mathbf{m}$ is the uniform tuple $(\max(\mathbf{d}), \ldots, \max(\mathbf{d}))$ of length $\widetilde{n}$. In terms of efficiency, it is more interesting to perform the reduction step on $\mathbf{X}^{\mathbf{m}-\mathbf{d}}\,\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ with the shift $-\mathbf{d}$, rather than on $\mathbf{A}$ with the shift $-\boldsymbol{\delta}$. Indeed, using the fastest known deterministic reduction algorithm [GSSV12], the latter computation uses $\mathcal{O}\tilde{}(n^{\omega}(\deg(\mathbf{A}) + \max(\boldsymbol{\delta})))$ field operations. On the other hand, the former is in $\mathcal{O}\tilde{}(n^{\omega}(\deg(\mathbf{A}) + \delta))$, since $\max(\mathbf{d}) \leqslant \delta$ and $\deg(\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})) \leqslant \deg(\mathbf{A})$. We recall that δ is close to the average of $\boldsymbol{\delta}$.

We state this formally in the following lemma. For the sake of presentation we postpone the proof until later in Section 16.4.4.

Lemma 16.19. *Let $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_n) \in \mathbb{Z}^n_{\geqslant 0}$, and define $\mathbf{d}$ as in Eq. (16.4).*

(i) If a matrix $\mathbf{R} \in \mathbb{K}[X]^{n\times n}$ is $-\boldsymbol{\delta}$-reduced with $-\boldsymbol{\delta}$-column degree $\mathbf{0}$, then $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R})$ is $-\mathbf{d}$-reduced with $-\mathbf{d}$-column degree $\mathbf{0}$.

(ii) If two matrices $\mathbf{A}$ and $\mathbf{B}$ in $\mathbb{K}[X]^{n\times n}$ are right unimodularly equivalent, then $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ and $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{B})$ are also right unimodularly equivalent.

(iii) If $\mathbf{A} \in \mathbb{K}[X]^{n\times n}$ is nonsingular, $\mathbf{R}$ is a $-\boldsymbol{\delta}$-reduced form of $\mathbf{A}$, and $\mathbf{R}$ has $-\boldsymbol{\delta}$-column degree $\mathbf{0}$, then $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R})$ is a $-\mathbf{d}$-reduced form of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ and the $-\mathbf{d}$-column degree of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R})$ is $\mathbf{0}$.

Our algorithm will first build $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ and then find a $-\mathbf{d}$-reduced form $\hat{\mathbf{R}}$ for this new matrix. We note that, for any $-\boldsymbol{\delta}$-reduced form $\mathbf{R}$ of $\mathbf{A}$, the matrix $\hat{\mathbf{R}} = \mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R})$ is a suitable reduced form and, as remarked earlier, has the property that it is easy to recover $\mathbf{R}$. However, it is not the case that any $\hat{\mathbf{R}}$ computed by shifted reduction from $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ will have the form $\hat{\mathbf{R}} = \mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R})$. In order to solve this issue, we will rely on normalization as in Corollary 16.16. This allows us to deduce $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H})$ from $\hat{\mathbf{R}}$, and then the entries of $\mathbf{H}$ can be read off from those of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H})$. Diagrammatically we have

$$\begin{array}{ccccc}
\mathbf{X}^{\mu-\boldsymbol{\delta}}\,\mathbf{A} & \xrightarrow{\text{reduction}} & \mathbf{X}^{\mu-\boldsymbol{\delta}}\,\mathbf{R} & \xrightarrow{\text{normalization}} & \mathbf{H} = \mathbf{R}\,\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})^{-1} \\
\Big| & & & & \Big| \\
\text{partial linearization} & & & & \text{partial linearization} \\
\downarrow & & & & \downarrow \\
\mathbf{X}^{\mathbf{m}-\mathbf{d}}\,\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A}) & \xrightarrow{\text{reduction}} & \mathbf{X}^{\mathbf{m}-\mathbf{d}}\,\hat{\mathbf{R}} & \xrightarrow{\text{normalization}} & \mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H}) = \hat{\mathbf{R}}\,\mathrm{lm}_{-\mathbf{d}}(\hat{\mathbf{R}})^{-1}
\end{array}.$$

Corollary 16.20. *Let $\mathbf{A} \in \mathbb{K}[X]^{n\times n}$ be nonsingular and let $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_n) \in \mathbb{Z}^n_{\geqslant 0}$ denote the degrees of the diagonal entries of the Hermite form $\mathbf{H}$ of $\mathbf{A}$. Using the notation from Definition 16.18, we have that*

(i) $\mathrm{lm}_{-\mathbf{d}}(\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H}))$ *is the identity matrix,*

(ii) if $\hat{\mathbf{R}} \in \mathbb{K}[X]^{\tilde{n}\times\tilde{n}}$ *is a* $-\mathbf{d}$*-reduced form of* $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$*, then* $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H}) = \hat{\mathbf{R}}\,\mathrm{lm}_{-\mathbf{d}}(\hat{\mathbf{R}})^{-1}$.

Proof. (i) follows from the construction of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H})$. From Lemma 16.19 we have that $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H})$ is a $-\mathbf{d}$-reduced form of $\mathbf{A}$, so that (ii) follows from (i) and Corollary 16.16. ☙

In particular, $\mathbf{H}$ can be recovered as being the submatrix of $\mathcal{E}\,\hat{\mathbf{R}}\,\mathrm{lm}_{-\mathbf{d}}(\hat{\mathbf{R}})^{-1}$ formed by its columns $\{\alpha_1 + \cdots + \alpha_i, 1 \leqslant i \leqslant n\}$.

Example 16.21 (Reducing the diagonal degrees). Consider a matrix $\mathbf{A} \in \mathbb{K}[X]^{4\times 4}$ such that its Hermite form $\mathbf{H}$ has diagonal degrees $\boldsymbol{\delta} = (2, 37, 7, 18)$. As shown earlier,

$$\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H}) = \begin{bmatrix} (2) & & & & & & \\ [16] & X^{17} & & [16] & & & \\ [16] & -1 & X^{17} & [16] & & & \\ [2] & & -1 & (3) & & & \\ [6] & & & [6] & (7) & & \\ [16] & & & [16] & [16] & X^{17} & [16] \\ [0] & & & [0] & [0] & -1 & (1) \end{bmatrix}.$$

We see that $\mathbf{d} = (2, 17, 17, 3, 7, 17, 1)$ corresponds to the row degree of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H})$, that this matrix has $-\mathbf{d}$-column degree $\mathbf{0}$ and that its $-\mathbf{d}$-leading matrix is the identity. In particular, it is $-\mathbf{d}$-reduced. In addition, from (ii) of Lemma 16.19, $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H})$ and $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ are right-unimodularly equivalent. As a result, $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H})$ is a $-\mathbf{d}$-reduced form of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$.

Let $\hat{\mathbf{R}}$ be any $-\mathbf{d}$-reduced form of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$. Then $\hat{\mathbf{R}}$ also has $-\mathbf{d}$-column degree $\mathbf{0}$, its $-\mathbf{d}$-leading matrix is invertible, and its degree profile is

$$\hat{\mathbf{R}} = \begin{bmatrix} [2] & [2] & [2] & [2] & [2] & [2] & [2] \\ [17] & [17] & [17] & [17] & [17] & [17] & [17] \\ [17] & [17] & [17] & [17] & [17] & [17] & [17] \\ [3] & [3] & [3] & [3] & [3] & [3] & [3] \\ [7] & [7] & [7] & [7] & [7] & [7] & [7] \\ [17] & [17] & [17] & [17] & [17] & [17] & [17] \\ [1] & [1] & [1] & [1] & [1] & [1] & [1] \end{bmatrix}.$$

While $\hat{\mathbf{R}}$ is generally not of the form $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R})$ for $\mathbf{R}$ some $-\boldsymbol{\delta}$-reduced form of $\mathbf{A}$, it still follows from Corollary 16.16 that $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H}) = \hat{\mathbf{R}}\,\mathrm{lm}_{-\mathbf{d}}(\hat{\mathbf{R}})^{-1}$. ✍

16.4.3 Algorithm and computational cost

The results in the previous subsection lead to Algorithm 39 for the computation of the Hermite form $\mathbf{H}$ from $\mathbf{A}$ and $\boldsymbol{\delta}$. Its main computational task is to compute a column reduced form of a matrix of dimension $\mathcal{O}(n)$ and degree $\mathcal{O}(\deg(\mathbf{A}))$ (Step **6**). This can be done efficiently and deterministically with the algorithm in [GSSV12, Section 8].

Proposition 16.22. *Let* $\mathbf{A} \in \mathbb{K}[X]^{n\times n}$ *be nonsingular, and let* $\boldsymbol{\delta} \in \mathbb{Z}^n_{\geqslant 0}$ *be the degrees of the diagonal entries of the Hermite form of* $\mathbf{A}$*. On input* $\mathbf{A}$ *and* $\boldsymbol{\delta}$*, Algorithm 39 computes the Hermite form of* $\mathbf{A}$ *using* $\mathcal{O}\tilde{}(n^\omega \deg(\mathbf{A}))$ *field operations.*

Proof. The correctness of the algorithm follows directly from Corollary 16.20 and from the remark that a matrix $\mathbf{R} \in \mathbb{K}[X]^{\widetilde{n}\times\widetilde{n}}$ is $-\mathbf{d}$-column reduced if and only if $\mathbf{DR}$ is column reduced (for the uniform shift), where $\mathbf{D}$ is the diagonal matrix at Step **5**.

Furthermore, we have $\deg(\mathbf{D}) \leqslant \delta$ and $\deg(\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})) \leqslant \max(\deg(\mathbf{A}), \delta)$. Since $\delta = 1 + \lfloor |\boldsymbol{\delta}|/n \rfloor$, and as $\mathbf{H}$ is in Hermite form and $\boldsymbol{\delta}$ are the degrees of its diagonal entries, we have $|\boldsymbol{\delta}| = \deg(\det(\mathbf{H})) = \deg(\det(\mathbf{A})) \leqslant n\deg(\mathbf{A})$. Thus, $\delta \leqslant 1 + \deg(\mathbf{A})$ and the degrees of $\mathbf{D}$ and $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ are both at most $1 + \deg(\mathbf{A})$. Their product $\mathbf{D}\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ therefore has degree at most $2 + 2\deg(\mathbf{A})$. On the other hand, these matrices have dimension

$$\widetilde{n} = \sum_{i=1}^{n} \alpha_i \leqslant \sum_{i=1}^{n} (1 + \delta_i/\delta) = n + \frac{|\boldsymbol{\delta}|}{1 + \lfloor |\boldsymbol{\delta}|/n \rfloor} < 2n.$$

As a result, Step **6** uses $\mathcal{O}\tilde{}(n^\omega \deg(\mathbf{A}))$ field operations [GSSV12, Theorem 18].

Concerning Step **7**, from Corollary 16.20 the matrix $\hat{\mathbf{R}}$ has row degree $\mathbf{d}$. Thus, since $\mathrm{lm}_{-\mathbf{d}}(\hat{\mathbf{R}})$ is a constant matrix, the computation of $\hat{\mathbf{R}}\mathrm{lm}_{-\mathbf{d}}(\hat{\mathbf{R}})^{-1}$ can be performed via complete linearization of the rows of $\hat{\mathbf{R}}$, using $\mathcal{O}(n^\omega \lceil |\mathbf{d}|/n \rceil)$ operations. This concludes the proof since $|\mathbf{d}| = |\boldsymbol{\delta}| = \deg(\det(\mathbf{H})) = \deg(\det(\mathbf{A})) \leqslant n\deg(\mathbf{A})$. ∎

Algorithm 39 – MinDegHermite
(Hermite form with known minimal degree)

Input:
- a nonsingular matrix $\mathbf{A} \in \mathbb{K}[X]^{n\times n}$,
- the diagonal degrees $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_n) \in \mathbb{Z}^n_{\geqslant 0}$ of the Hermite form of $\mathbf{A}$.

Output: the Hermite form of $\mathbf{A}$.

1. $\delta \leftarrow 1 + \lfloor (\delta_1 + \cdots + \delta_n)/n \rfloor$

2. For i from 1 to n

  a. If $\delta_i > 0$: $\alpha_i \leftarrow \lceil \delta/\delta_i \rceil$; $\beta_i \leftarrow \delta_i - (\alpha_i - 1)\delta$

  b. Else: $\alpha_i \leftarrow 1$; $\beta_i \leftarrow 0$;

3. $\widetilde{n} \leftarrow \alpha_1 + \cdots + \alpha_n$ and $\mathcal{E} \in \mathbb{K}^{\widetilde{n}\times n}$ as in Eq. (16.5)

4. $\mathbf{d} \leftarrow (d_1, \ldots, d_{\widetilde{n}})$ as in Eq. (16.4)

5. $\mathbf{D} \leftarrow \mathrm{Diag}(X^{\delta - d_1}, \ldots, X^{\delta - d_{\widetilde{n}}})$

6. $\mathbf{D}\hat{\mathbf{R}} \leftarrow \mathrm{ColumnReduce}(\mathbf{D}\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A}))$ `// algorithm [GSSV12, Fig.7]`

7. $\hat{\mathbf{H}} \leftarrow \mathcal{E}\,\hat{\mathbf{R}}\,\mathrm{lm}_{-\mathbf{d}}(\hat{\mathbf{R}})^{-1}$

8. $\mathbf{H} \leftarrow$ the submatrix of $\hat{\mathbf{H}}$ formed by its columns $\{\alpha_1 + \cdots + \alpha_i, 1 \leqslant i \leqslant n\}$

9. Return $\mathbf{H}$

Combining Algorithms 37 and 39 results in a deterministic algorithm for computing the Hermite form of a nonsingular matrix $\mathbf{A}$ in $\mathcal{O}\tilde{}(n^\omega \deg(\mathbf{A}))$ field operations. This is

the main step towards the proof of our result announced in Theorem 3.9; it remains to deal with the possible non-uniformity of the degrees in $\mathbf{A}$.

Remark 16.23. At the time of writing, detailed cost bounds with logarithmic factors are available for all operations in the algorithm except the column basis computation at Step **2.b** of Algorithm 37. Analyzing this cost would directly yield a cost bound with logarithmic factors for Hermite form and determinant computation. ☕

Example 16.24. Let $\mathbb{K} = \mathbb{Z}_7$ be the field with 7 elements, and consider the matrix $\mathbf{A} \in \mathbb{K}[X]^{3\times 3}$ from Example 16.5:

$$\mathbf{A} = \begin{bmatrix} 6X+1 & 2X^3+X^2+6X+1 & 3 \\ 4X^5+5X^4+4X^2+X & 6X^5+5X^4+2X^3+4 & X^4+5X^3+6X^2+5X \\ 2 & 2X^5+5X^4+5X^3+6X^2 & 6 \end{bmatrix}.$$

According to Example 16.5 the diagonal entries of the Hermite form of $\mathbf{A}$ have degrees $\boldsymbol{\delta} = (0, 1, 9)$. Note that $\boldsymbol{\delta}$ is non-uniform, and $\max(\boldsymbol{\delta}) - \min(\boldsymbol{\delta}) = \deg(\det(\mathbf{A})) - 1$.

Using the column reduction algorithm in [GSSV12] to compute a $-\boldsymbol{\delta}$-reduced form of $\mathbf{A}$ would imply working on the matrix $\mathbf{X}^{\mu-\boldsymbol{\delta}}\,\mathbf{A} = \mathbf{X}^{(9,8,0)}\,\mathbf{A}$, which has degree $13 = \deg(\det(\mathbf{A})) + \deg(\mathbf{A}) - 2$. In this case partial linearization gives us a 5×5 matrix $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ and a shift $\mathbf{d}$ such that $\deg(\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})) \leqslant \deg(\mathbf{A})$ and $\max(\mathbf{d}) - \min(\mathbf{d}) \leqslant \deg(\mathbf{A})$. In particular, the matrix $\mathbf{X}^{\mathbf{m}-\mathbf{d}}\,\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ to be reduced has degree $8 \leqslant 2\deg(\mathbf{A})$.

To see this, Definition 16.18 gives the parameters $\delta = 4$, $\alpha = (1, 1, 3)$, $\beta = (0, 1, 1)$, $\mathbf{d} = (0, 1, 4, 4, 1)$, the expansion-compression matrix

$$\mathcal{E} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & X^4 & X^8 \end{bmatrix},$$

and finally

$$\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A}) = \begin{bmatrix} 6X+1 & 2X^3+X^2+6X+1 & 0 & 0 & 3 \\ 4X^5+5X^4+4X^2+X & 6X^5+5X^4+2X^3+4 & 0 & 0 & X^4+5X^3+6X^2+5X \\ 2 & 5X^3+6X^2 & X^4 & 0 & 6 \\ 0 & 2X+5 & 6 & X^4 & 0 \\ 0 & 0 & 0 & 6 & 0 \end{bmatrix}.$$

Computing a $-\mathbf{d}$-reduced form for $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ gives

$$\hat{\mathbf{R}} = \begin{bmatrix} 5 & 1 & 0 & 1 & 2 \\ 5 & 4X+4 & 0 & 3X+5 & 6X+3 \\ X^3+6X^2+4 & 3X^4+X^3+6X^2 & X^4 & X^3+5X^2+4X+3 & 6X^4+2X^3+3X^2+X+6 \\ 3X^3+4X^2+6 & 4X^4+4X^3+4X+5 & 6 & X^3+2X+4 & 5X^4+2X^3+4X+2 \\ 6 & X & 0 & 6 & 0 \end{bmatrix}.$$

Note that $\mathrm{rdeg}(\mathbf{R}) = \mathbf{d}$, and more precisely,

$$\mathrm{lm}_{-\mathbf{d}}(\mathbf{R}) = \begin{bmatrix} 5 & 1 & 0 & 1 & 2 \\ 0 & 4 & 0 & 3 & 6 \\ 0 & 3 & 1 & 0 & 6 \\ 0 & 4 & 0 & 0 & 5 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

Normalizing $\hat{\mathbf{R}}$ via $\hat{\mathbf{R}}\mathrm{lm}_{-\mathbf{d}}(\mathbf{R})^{-1}$ gives

$$\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H}) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & X+6 & 0 & 0 & 0 \\ 3X^3+4X^2+5 & 4X^3+5X^2+6X+4 & X^4 & 0 & 3X^3+3X^2+4X \\ 2X^3+5X^2+4 & 2X^3+3X^2+3X & 6 & X^4 & X^3+4X^2+6X+4 \\ 4 & 3 & 0 & 6 & X+2 \end{bmatrix}.$$

Performing the inverse linearization, by taking columns $(1,2,5)$ of $\mathcal{E}\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{H})$, directly gives the entries in the Hermite form of $\mathbf{A}$:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & X+6 & 0 \\ h_{31} & h_{32} & X^9+2X^8+X^7+4X^6+6X^5+4X^4+3X^3+3X^2+4X \end{bmatrix}$$

with

$$\begin{aligned} h_{31} &= 4x^8+2x^7+5x^6+4x^4+3x^3+4x^2+5, \\ h_{32} &= 3x^8+2x^7+3x^6+3x^5+4x^3+5x^2+6x+4. \end{aligned}$$ ✍

16.4.4 Proof of Lemma 16.19

Let us now give the detailed proof of Lemma 16.19.

(i) Since $\mathbf{R} \in \mathbb{K}[X]^{n\times n}$ is $-\boldsymbol{\delta}$-reduced with $-\boldsymbol{\delta}$-column degree $\mathbf{0}$, it has row degree $\boldsymbol{\delta}$ since otherwise the invertible matrix $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})$ would have a zero row. We show that $\mathrm{lm}_{-\mathbf{d}}(\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R}))$ is a permutation of the rows and columns of $\begin{bmatrix} \mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R}) & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \in \mathbb{K}^{\tilde{n}\times\tilde{n}}$. In particular, $\mathrm{lm}_{-\mathbf{d}}(\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R}))$ is invertible and thus $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R})$ is $-\mathbf{d}$-reduced.

Let us first observe it on an example. We consider the case $\boldsymbol{\delta} = (2,37,7,18)$. Then $\mathbf{R}$ has the following degree profile,

$$\mathbf{R} = \begin{bmatrix} [2] & [2] & [2] & [2] \\ [37] & [37] & [37] & [37] \\ [7] & [7] & [7] & [7] \\ [18] & [18] & [18] & [18] \end{bmatrix}$$

with invertible $-\boldsymbol{\delta}$-leading matrix. Following the construction in Definition 16.18, we have $\mathbf{d} = (2,17,17,3,7,17,1)$ and

$$\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R}) = \begin{bmatrix} [2] & & & [2] & [2] & & [2] \\ [16] & X^{17} & & [16] & [16] & & [16] \\ [16] & -1 & X^{17} & [16] & [16] & & [16] \\ [3] & & -1 & [3] & [3] & & [3] \\ [7] & & & [7] & [7] & & [7] \\ [16] & & & [16] & [16] & X^{17} & [16] \\ [1] & & & [1] & [1] & -1 & [1] \end{bmatrix}.$$

Observe that $\mathbf{R}$ has $-\mathbf{d}$-column degree at most $\mathbf{0}$ componentwise, and that its $-\mathbf{d}$-leading matrix is

$$\mathrm{lm}_{-\mathbf{d}}(\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R})) = \begin{bmatrix} \ell_{11} & & & \ell_{12} & \ell_{13} & & \ell_{14} \\ & 1 & & & & & \\ & & 1 & & & & \\ \ell_{21} & & & \ell_{22} & \ell_{23} & & \ell_{24} \\ \ell_{31} & & & \ell_{32} & \ell_{33} & & \ell_{34} \\ & & & & & 1 & \\ \ell_{41} & & & \ell_{42} & \ell_{42} & & \ell_{42} \end{bmatrix},$$

where $(\ell_{ij})_{1\leqslant i,j\leqslant 4} = \mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})$. Since $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})$ is invertible, $\mathrm{lm}_{-\mathbf{d}}(\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R}))$ is invertible as well. Furthermore $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R})$ is $-\mathbf{d}$-reduced and that it has $-\mathbf{d}$-column degree $\mathbf{0}$.

In the general case, by construction of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R})$ one can check that $\mathrm{lm}_{-\mathbf{d}}(\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R}))$ is a matrix in $\mathbb{K}^{\widetilde{n}\times\widetilde{n}}$ such that

- its $n \times n$ submatrix with row and column indices in $\{\alpha_1 + \cdots + \alpha_i, 1 \leqslant i \leqslant n\}$ is equal to $\mathrm{lm}_{-\boldsymbol{\delta}}(\mathbf{R})$,
- its $(\widetilde{n} - n) \times (\widetilde{n} - n)$ submatrix with row and column indices in $\{1, \ldots, \widetilde{n}\} - \{\alpha_1 + \cdots + \alpha_i, 1 \leqslant i \leqslant n\}$ is equal to the identity matrix,
- its other entries are all zero.

This directly implies that $\mathrm{lm}_{-\mathbf{d}}(\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R}))$ is invertible. In addition by construction $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R})$ has $-\mathbf{d}$-column degree at most $\mathbf{0}$ componentwise. The fact that $\mathrm{lm}_{-\mathbf{d}}(\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R}))$ is invertible also implies that $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{R})$ has $-\mathbf{d}$-column degree exactly $\mathbf{0}$.

(ii) Denote by $\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}} \in \mathbb{K}[X]^{\widetilde{n}\times(\widetilde{n}-n)}$ the submatrix of $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ formed by its columns at indices $\{\alpha_1 + \cdots + \alpha_i + j, 1 \leqslant j \leqslant \alpha_{i+1} - 1, 0 \leqslant i \leqslant n-1\}$. Up to a permutation of its columns, $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ is then $[\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}} \;\; \widetilde{\mathbf{A}}]$. In particular, $\mathcal{E}\,\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$ is right-unimodularly equivalent to $\mathcal{E}\,[\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}} \;\; \widetilde{\mathbf{A}}] = [\mathbf{0} \;\; \mathbf{A}]$. For the remainder of this proof we will use the shorthand notation $\mathcal{E}\,\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A}) \equiv [\mathbf{0} \;\; \mathbf{A}]$.

Define the matrix $\mathbf{E} \in \mathbb{K}^{(\widetilde{n}-n)\times\widetilde{n}}$ whose row $\alpha_1 + \cdots + \alpha_i + j - i$ is the coordinate vector with 1 at index $\alpha_1 + \cdots + \alpha_i + j + 1$, for all $1 \leqslant j \leqslant \alpha_{i+1} - 1$ and $0 \leqslant i \leqslant n-1$. That is, we have

$$\begin{bmatrix} \mathbf{E} \\ \mathcal{E} \end{bmatrix} = \begin{bmatrix} 0 & 1 & & & & & & & & \\ & & \ddots & & & & & & & \\ & & & & 1 & & & & & \\ & & & & & \ddots & & & & \\ & & & & & & 0 & 1 & & \\ & & & & & & & & \ddots & \\ & & & & & & & & & 1 \\ 1 & X^{\delta} & \cdots & X^{(\alpha_1-1)\delta} & & & & & & \\ & & & & & \ddots & & & & \\ & & & & & & 1 & X^{\delta} & \cdots & X^{(\alpha_n-1)\delta} \end{bmatrix}.$$

By construction, the matrix $\mathbf{U} = \mathbf{E}\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}}$ is upper triangular with diagonal entries -1, and thus unimodular. As a result,

$$\begin{bmatrix}\mathbf{E}\\ \mathcal{E}\end{bmatrix}\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A}) \equiv \begin{bmatrix}\mathbf{E}\\ \mathcal{E}\end{bmatrix}\begin{bmatrix}\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}} & \widetilde{\mathbf{A}}\end{bmatrix} = \begin{bmatrix}\mathbf{U} & *\\ \mathbf{0} & \mathbf{A}\end{bmatrix} \equiv \begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{A}\end{bmatrix}.$$

Similarly, we have that $\begin{bmatrix}\mathbf{E}\\ \mathcal{E}\end{bmatrix}\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{B}) \equiv \begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{B}\end{bmatrix}$.

Since $\mathbf{A} \equiv \mathbf{B}$ by assumption, we obtain $\begin{bmatrix}\mathbf{E}\\ \mathcal{E}\end{bmatrix}\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A}) \equiv \begin{bmatrix}\mathbf{E}\\ \mathcal{E}\end{bmatrix}\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{B})$. This implies that $\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A}) \equiv \mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{B})$ since the matrix $\begin{bmatrix}\mathbf{E}\\ \mathcal{E}\end{bmatrix}$ is invertible (more precisely, its determinant is 1).

(iii) is a direct consequence of (i) and (ii).

16.5 Reduction to almost uniform input degrees

We have already explained in Section 15.1 our interest in obtaining a cost bound which involves the generic determinant bound. In Section 16.2 we showed how to compute the diagonal entries of $\mathbf{H}$ in $\tilde{\mathcal{O}}(n^\omega\lceil s\rceil)$ operations, with s the average column degree of the input matrix. However, this does not take into account the fact that the degrees of its rows are possibly unbalanced. Besides, in Section 16.4, we were only able to obtain the cost bound $\tilde{\mathcal{O}}(n^\omega \deg(\mathbf{A}))$ for computing the remaining entries of $\mathbf{H}$.

The results in Section 15.2 allow us to reduce from the general case of Hermite form computation to the case where the degree of the input matrix $\mathbf{A}$ is in $\mathcal{O}(\lceil D_{\mathbf{A}}/n\rceil)$, while the dimension of the matrix $\mathbf{A}$ remains in $\mathcal{O}(n)$. Combined with Propositions 16.6 and 16.22, this proves Theorem 3.9.

We remark that Hermite forms in Section 15.2 are under left-unimodular equivalence, while here we work with right-unimodular equivalence. Furthermore, it is not only a matter of transposing matrices, since despite not being for the same kind of unimodular equivalences, both in Section 15.2 and here the Hermite form refers to a lower triangular form. Of course, the partial linearization techniques can still be applied, but this requires a slight modification of the definitions in order to adapt the results of Section 15.2 to the context here. Roughly, one will permute positions of the blocks of rows and columns so that the sought Hermite form appears as the *trailing* principal submatrix, instead of the leading one. The complete details of these modified definitions can be found in [LNZ16, Section 5]; we only show it on an example below.

Example 16.25. Let $\mathbb{K} = \mathbb{Z}_7$ be the field with 7 elements, and consider the matrix $\mathbf{A} \in \mathbb{K}[X]^{3\times 3}$ from Example 16.5:

$$\mathbf{A} = \begin{bmatrix} 6X+1 & 2X^3+X^2+6X+1 & 3\\ 4X^5+5X^4+4X^2+X & 6X^5+5X^4+2X^3+4 & X^4+5X^3+6X^2+5X\\ 2 & 2X^5+5X^4+5X^3+6X^2 & 6\end{bmatrix}.$$

After sorting the triples we obtain the linearization degrees $\boldsymbol{\delta} = (1, 5, 0)$ resulting in

$$\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A}) = \begin{bmatrix} 1 & 0 & 6X^3 & 0 \\ 2 & 6X+1 & X^2+6X+1 & 3 \\ 6X^2+5X+2 & 4X^5+5X^4+4X^2+X & 4 & X^4+5X^3+6X^2+5X \\ 2X^2+5X+5 & 2 & 6X^2 & 6 \end{bmatrix},$$

and $\gamma = \operatorname{rdeg}(\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})) = (3, 2, 5, 2)$ giving

$$\mathcal{L}^{\mathrm{r}}_{\gamma}(\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})) = \begin{bmatrix} 1 & 0 & 4x+5 & 0 & 1 \\ 0 & 1 & 0 & 6x^3 & 0 \\ 0 & 2 & 6x+1 & x^2+6x+1 & 3 \\ 6x^4 & 6x^2+5x+2 & 4x^2+x & 4 & 5x^3+6x^2+5x \\ 0 & 2x^2+5x+5 & 2 & 6x^2 & 6 \end{bmatrix}.$$

Computing the diagonal entries of the Hermite form of $\mathcal{L}^{\mathrm{r}}_{\gamma}(\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A}))$ as in Section 16.2, we obtain their degrees $(0, 0, 0, 1, 9)$. Proceeding as in Section 16.4 we can to compute the complete Hermite form using the knowledge of these degrees giving

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{R} & \mathbf{H} \end{bmatrix}$$

with

$$\mathbf{R} = \begin{bmatrix} 0 & 0 \\ 6 & 4 \\ X^7+4X^6+2X^4+3X^3+5X^2+3X+4 & 4X^8+2X^7+4X^6+5X^5+5X^4+2X^3+6X^2+X \end{bmatrix}$$

and $\mathbf{H}$ the Hermite form of $\mathbf{A}$ as given in Example 16.24. ✍

Perspectives

We conclude this document by presenting some of our perspectives for future work on the problems tackled in this thesis.

⋆⋆⋆

Fast change of monomial order

Our results in Chapters 4 and 5 lead to a fast change of monomial order algorithm for zero-dimensional ideals in $\mathbb{K}[X_1, \ldots, X_r]$, under the assumption that the initial ideal is Borel-fixed (see Section 2.2.3). This is achieved by following a two-step approach: we first determine the multiplication matrices from the input Gröbner basis, and then we use them to compute the Gröbner relation basis for the target monomial order. For an ideal of degree D, we obtained the cost bound $\tilde{\mathcal{O}}(rD^\omega)$ (Theorem 2.14).

A first extension of this result would be to support the case of a *module* which has finite codimension D as a $\mathbb{K}$-vector space, with a similar cost bound. We remark that the second step already deals with this more general situation (see Sections 2.1 and 2.2). Then, it remains to study whether some Borel-fixedness property of the initial module would allow us to use an approach similar to that in Chapter 5 to efficiently retrieve the multiplication matrices from the input Gröbner basis.

Another, more significant improvement would be to remove the assumption that the initial ideal is Borel-fixed. We note that we only use this assumption in the first step of our algorithm. Hence the question: how to compute the multiplication matrices, without assumption on the ideal, using $\tilde{\mathcal{O}}(rD^\omega)$ operations? To the best of our knowledge, this is not known as of today, even for specific monomial orders. The case of the degree-reverse lexicographic order has been studied in [FGHR13], with a cost bound of $\tilde{\mathcal{O}}(\delta r^\omega D^\omega)$ operations, where δ is related to the degrees in the input Gröbner basis; how to remove this factor δ is unclear to us.

Finally, we would like to study more in depth how our results for the computation of Gröbner relation bases compare to the different situations that are handled in [MMM93].

⋆⋆⋆

Systems of linear modular equations

In the computation of shifted Popov solution bases for systems of linear modular univariate equations, we assumed that the number of equations was bounded from above by the number of unknowns; let $\mathcal{H}$ denote this assumption, namely $n \in \mathcal{O}(m)$ (Theorem 2.22).

On the other hand, in the specific case where the moduli split and are given by their roots and multiplicities, we managed to remove this assumption (Theorem 2.20). One

motivation was that $\mathcal{H}$ was not satisfied in the application to multivariate interpolation problems arising in the soft-decoding of Reed-Solomon codes. Then, a natural perspective is to aim at a similar result when the moduli are arbitrary polynomials given by their coefficients; or in other words, at removing the assumption $\mathcal{H}$ in Theorem 2.22.

The two main uses of $\mathcal{H}$ were in the computations of the residual and of the solution basis when we have a priori degree information. Concerning the former, we expect that $\mathcal{H}$ can be avoided without impacting the cost beyond logarithmic factors, by following an approach based on Chinese remaindering techniques similar to the computation of the residual for moduli which split with known roots (Section 14.2). As for finding the solution basis when the minimal degree is known, for the moment it is not clear to us how to remove the assumption $\mathcal{H}$; an obstacle with our current approach is that it resorts to computing a left kernel basis of a polynomial matrix which has $m + n$ rows.

⋆ ⋆ ⋆

Kernel bases of polynomial matrices

The above-mentioned algorithm for solution bases led us to design an efficient algorithm for computing the shifted Popov kernel basis of a polynomial matrix, when information about pivots is known a priori (Proposition 8.6). In this context, and unlike most of this thesis, we are considering rectangular shifted Popov forms, corresponding to submodules of $\mathbb{K}[X]^m$ of rank strictly less than m; in addition to knowing the degrees of the pivots, we also require the knowledge of the indices of the columns containing the pivots.

During a research internship of Vũ Thị Xuân, co-supervised with Claude-Pierre Jeannerod, we explored further the question of computing shifted Popov kernel bases for arbitrary shifts. We showed that, with one call to our solution basis algorithm for a single modular equation and to our approximant basis algorithm, one can compute the shifted Popov kernel basis of a $m \times 1$ column vector of degree D in $\mathcal{O}\tilde{}(m^{\omega-1}D)$ operations. Compared to [ZLS12], where a similar cost bound was achieved, this new algorithm supports arbitrary shifts and returns the basis in normal form.

Furthermore, Xuân proved a generalization of Theorem 1.28 in the rectangular case; for kernel bases, this means that one can directly retrieve pivot information of the sought basis by reading the degrees and indices of the pivots of two bases computed recursively, for example in a divide-and-conquer scheme on the columns of the input $m \times n$ matrix as in [ZLS12]. Having this information, the basis can be obtained efficiently (Proposition 8.6).

Thus, we have a divide-and-conquer scheme with a fast base case for a column vector, and an efficient way of combining the results of recursive calls. Yet, as such, these ingredients do not lead to a fast algorithm for shifted Popov kernel bases in general, because of degree growth in the computation of residuals. We will continue to work on this project.

⋆ ⋆ ⋆

Normal forms of polynomial matrices

We have given a fast algorithm for computing the shifted Popov normal form of a polynomial matrix, for an arbitrary shift (Theorem 3.7). This algorithm is probabilistic, requires that the matrix be square and nonsingular, and uses $\mathcal{O}\tilde{}(m^{\omega}d)$ field operations where m and d are the row dimension and the degree of the input matrix.

We would like to find a similarly efficient algorithm which is deterministic. A solution is to design a deterministic Smith form algorithm in $\tilde{\mathcal{O}}(m^\omega d)$, whereas here we rely on the probabilistic one of [Sto03]. Another one is to compute the diagonal degrees of the shifted Popov form without finding the whole matrix; then, Theorem 3.10 gives a reduction to the case of the uniform shift, which is dealt with deterministically in [SS11, GSSV12]. While this approach was successful for the Hermite form (Chapter 16), where we exploited its triangular shape, it is unclear to us how to proceed for an arbitrary shift.

Focusing on the important case of the uniform shift, could we rely on [SS11, GSSV12] to deterministically compute the Popov form in $\tilde{\mathcal{O}}(m^\omega \lceil D/m \rceil)$ operations, where D is the generic determinant bound? Until now, this cost bound involving average degrees has been obtained for the probabilistic computation of shifted Popov forms and for the deterministic computation of the Hermite form. For this question, we also refer to Remark 3.6.

Furthermore, we would like to study the case of shifted Popov forms of matrices that are rectangular with an arbitrary rank. In this context, the row space basis algorithm of [ZL13] can be used to reduce to the case of full row rank. Then, in the latter case, how to efficiently compute the shifted Popov form? Is there a way to efficiently find, without computing the whole form, which columns contain pivots and what their degrees are?

⋆ ⋆ ⋆

Multivariate problems and multi-level structure

In this thesis, we have studied systems of linear modular equations for arbitrary system matrices $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ and normal forms of arbitrary nonsingular matrices $\mathbf{A} \in \mathbb{K}^{m\times m}$. However, when one faces these questions in specific contexts, these matrices often bear some type of structure which might be exploited for further efficiency.

Hereafter, for the sake of presentation, we focus on the interpolation step of the Sudan list-decoding algorithm for Reed-Solomon codes. Similar remarks hold for the more general list-decoding algorithms and the Coppersmith technique over $\mathbb{K}[X]$ mentioned in this document, and also for other applications that we did not study here such as the use of Hermite-Padé approximation for guessing linear differential equations.

In the Sudan decoding, we are given points $\{(x_1, y_1), \ldots, (x_D, y_D)\}$ with distinct x_i's, a degree bound m, and we look for an element of small shifted degree in the $\mathbb{K}[X]$-module of polynomials $Q \in \mathbb{K}[X, Y]$ such that $\deg_Y(Q) < m$ and $Q(x_i, y_i) = 0$ for $1 \leqslant i \leqslant D$.

Writing $Q = Q_0(X) + Q_1(X)Y + \cdots + Q_{m-1}(X)Y^{m-1}$, this can be done by finding a shifted reduced solution basis for the linear modular equation

$$Q_0 + Q_1 L + \cdots + Q_{m-1} L^{m-1} = 0 \bmod M$$

where L is the interpolant of degree less than D such that $L(x_i) = y_i$ for all i, and M is the product $\prod_i (X - x_i)$. Note that the left-hand side is $Q(X, L(X))$. Here, the system matrix is $\mathbf{F} = [1 \;\; L \;\; \cdots \;\; L^{m-1}]^\mathsf{T}$. In the lattice-based approach (see Sections 3.1 and 11.3), one rather computes a shifted reduced form of a basis of the module, chosen as

$$\mathbf{A} = \begin{bmatrix} M & & & \\ -L & 1 & & \\ \vdots & & \ddots & \\ -L^{m-1} & & & 1 \end{bmatrix} \quad \text{or} \quad \mathbf{A} = \begin{bmatrix} M & & & \\ -L & 1 & & \\ & \ddots & \ddots & \\ & & -L & 1 \end{bmatrix}.$$

Both $\mathbf{F}$ and $\mathbf{A}$ have some structure, which results from the bivariate nature of the initial problem. Note in particular that these matrices can be represented concisely with $\mathcal{O}(D)$ field elements; furthermore, we seek a single small shifted degree vector Q whose dense representation uses $\mathcal{O}(D)$ elements. The fastest known algorithms do not exploit this structure, and their cost bound is $\mathcal{O}\tilde{}(m^{\omega-1}D)$ while one might hope for $\mathcal{O}\tilde{}(D)$.

To put this in perspective, we remark that this question is close to the tasks of computing bivariate evaluation and modular composition, which have been studied in [NZ04, KU11], and for which no algorithm with quasi-linear cost bound over an abstract field are known, to the best of our knowledge. More precisely, verifying that some bivariate polynomial Q is a solution to our problem is equivalent to verifying that the bivariate evaluations $Q(x_i, y_i)$ are zero for all i; since the x_i's are distinct, this is also equivalent to verifying that the modular identity $Q(X, L(X)) = 0 \bmod M$ holds.

In this context, a natural question is whether we could take advantage of the algorithms in [NZ04, KU11] to exploit the above-mentioned structure. In addition, we would like to establish problem reductions in order to better identify the conceptual difficulties towards further improvements over the existing fast algorithms.

From a linear algebra viewpoint, this bivariate interpolation problem can be seen as a linear system over $\mathbb{K}$. Considering the multiplication matrices $\mathbf{M}_X = \mathrm{diag}(x_1, \ldots, x_D)$ and $\mathbf{M}_Y = \mathrm{diag}(y_1, \ldots, y_D)$, finding Q amounts to computing a vector in the left nullspace of the multi-Krylov matrix whose rows are $[1 \ \cdots \ 1]\mathbf{M}_X^i\mathbf{M}_Y^j$, where $0 \leqslant j < m$ and the range of the exponent i depends on the degree constraints specified by the shift. Thus, here the multi-Krylov matrix has a two-level Vandermonde structure.

More generally, problems of interpolation with multiplicities in r variables can be reduced to a linear system over $\mathbb{K}$, where the matrix of the system has an r-level block-Vandermonde structure. In this thesis, we solved such interpolation problems by reducing them to problems about modules over the univariate polynomials, and we ignored the structure of the input in the obtained instances of the latter problems. In other words, we have used and designed algorithms that only consider a single level of structure.

To the best of our knowledge, it is currently unknown how to solve such multi-level structured linear systems faster than by considering them as structured systems with a single level, ignoring the others. Again, multivariate evaluation corresponds to computing a matrix-vector product for a multi-level Vandermonde matrix: may the work of [NZ04, KU11] be used to speed up the solving of multi-level Vandermonde linear systems?

This question is not confined to multivariate interpolation and a structure of the Vandermonde type. For example, a similar situation occurs in multivariate generalizations of the Berlekamp-Massey algorithm [Sak88, Sak90]: in [BBF15, BBF17], only one level of structure of the multi-Hankel matrix arising in this context is exploited.

Index

Mathematical notations can be found at the end of this index.

approximant, **70**
approximant basis, 8, **70**
assumption
  $\mathcal{H}_{\text{int},1}$, **92**
  $\mathcal{H}_{\text{int},2}$, **92**
  $\mathcal{H}_{\text{int},3}$, **92**
  $\mathcal{H}_{\text{int},4}$, **92**
  $\mathcal{H}_{\boldsymbol{\beta}}$, 65, **127**
  $\mathcal{H}_{\mathsf{MM}(\cdot,\cdot)}$, **156**
  $\mathcal{H}_{\mathsf{M}(\cdot)}$, **155**
  $\mathcal{H}_{\mathbf{s},1}$, **37**
  $\mathcal{H}_{\mathbf{s},2}$, **37**

Beckermann-Labahn algorithm [BL94], 5, 78
Berlekamp-Massey algorithm, 2
Berlekamp-Massey-Sakata algorithm, 324
bivariate interpolation, **22**, **323**
border (of monomial basis), **52**, 142
Borel-fixed, 69, **142**

Cauchy interpolation, 2
change of monomial order, 54, **68**
change of shift, 9, **265**
characteristic polynomial, 5
Chinese remaindering, 8, **269**
column basis, **298**
coordinate vector, **31**, **41**
Coppersmith technique, 101
Coppersmith technique over $\mathbb{K}[X]$, **102**
  interpolation step, **103**

degree
  column degree, **23**
  of a matrix, 23
  of a vector, 23
  row degree, **23**
  shifted column degree, **23**
  shifted row degree, **23**
degree reverse lexicographic order, **43**
division with remainder, **31**, 46

Euclidean algorithm, 2
extended key equations, 87, **205**

FGLM algorithm, 7, 68
folded Reed-Solomon code, 91
full rank (polynomial matrix), 22

generic determinant bound, 107, **286**
generic initial ideal, 142
Goppa codes decoding, 2
Gröbner basis, 7, 20, 39, **46**
  minimal, **47**
  reduced, 20, 39, **47**
Gröbner relation basis, **58**
Guruswami-Rudra algorithm, 92
Guruswami-Sudan algorithm, 91–93

Half-gcd algorithm, 3, 78
Hasse derivative, 91, **230**
Hermite form, 5, 20, 27, **29**, **106**
Hermite-Padé approximation, 22, 70
Hilbert Basis Theorem, 38, **40**
hypothesis, *see* assumption

ideal, 20
identity matrix, **21**
indexing function, **122**
initial module, **44**
initial term, **44**
interpolant, **75**
interpolant basis, **75**
interpolation step, **102, 103**

Kötter's algorithm, 95
Kötter-Vardy algorithm, 93
Keller-Gehrig algorithm, 5
kernel (of a polynomial matrix), 22, **186**
kernel basis, **186**
key equation, 2, 87
Knuth-Schönhage-Moenck algorithm, 3
Krylov matrix, 4

Lagrange interpolation, 8
leading matrix, **24**
s-leading matrix, **24**
lexicographic order, **43**
linear functional, **59**, 98
linear modular equation, 2
linearization, 119, 120
linearly recurrent sequence, 2, 324
list-decoding
    folded Reed-Solomon codes, 89, 92, 95
    Parvaresh-Vardy codes, 91, 95
    Reed-Solomon codes, 89, 92
list-size condition, **90**

M-Padé approximant, 75
M-Padé approximation, 95
minimal degree, 19, **30**
    s-minimal degree, **30**
    shifted minimal degree, **30**
minimal Gröbner basis, *see* Gröbner basis
module, 19
    finite codimension, 31, 39, **51**
    finite-dimensional, 20, 31, 39, 51
    free, **20**
    Gröbner basis, 19
    rank, **20**
$\mathbb{K}[X]$-module, 19
$\mathbb{K}[\boldsymbol{X}]$-module, 38
monomial, **41**
    divisible, **41**
monomial basis, 31, 41, **45**
monomial ideal, **41**
monomial order, 20, 39, **43**
monomial submodule, **41**
    minimal generating set, **41**
multi-Krylov matrix, 65, 119, **124**
multiplication matrix, 3, 20, 39, **51**
multiplication table, 53
multiplicity, *see* root with mult., 90
multipoint Padé approximation, 2
multivariate interpolation, 89
    with multiplicities, 90, 96

Newton Padé approximation, 2
nonsingular, 22
normal form $\mathrm{nf}_{\prec}(f)$, **46**

order basis, 70
ordered weak Popov
    s-ordered weak Popov form, **27**
    shifted ordered weak Popov form, **27**
ordered weak Popov form, **27**

Padé approximation, 2
Parvaresh-Vardy code, 91
pivot
    s-pivot degree, **27, 28**
    s-pivot entry, **27**
    s-pivot index, **27, 28**
PM-Basis algorithm [GJV03], 78
polynomial matrix, 20, **21**
Popov basis, *see also* Popov form, **29**
    s-Popov basis, **29**
Popov form, 5, 19, 20, 27, **29**
    s-Popov form, **29**
    shifted Popov form, 19, **29**
position-over-term, **43**
POT order, **44**
predictable degree property, **25**
Private Information Retrieval, 89, 95

quasi Popov, *see* ordered weak Popov
quotient module, 31

rank, **20**, 22
rational function reconstruction, **2**
multivariate rational reconstruction, 40
Re-encoding technique, 93, 101, 235
reduced form, 19, 20, **24**
    shifted reduced form, 19, **24**
reduced Gröbner basis, *see* Gröbner basis
Reed-Solomon code, 91

Reed-Solomon codes, 2
relation, 7, **57**, **60**
relation basis, **58**, **61**
residual, 3, 5, 78, 158, 268
root with multiplicity, **90**
root with support, **97**
root-finding step, **102**
row rank profile, **127**
row space, **21**, 22

shift, 5, 19, 20, **23**
    uniform shift, **23**
shift matrix, **24**
shifted degree, 20, **23**
s-degree, **23**
shifted minimal relation basis, **61**
shifted Popov form, 27, **106**
shifted Popov relation basis, **61**
s-row degree, **23**
shifted TOP order, **44**, **49**
σ-basis, 70
soft-decoding, 89, 91, 93, 96
solution, **81**
solution basis, **81**
standard representation (Jordan matrix), **77**
Sudan list-decoding, **93**, **323**
support, *see* root with support
system of linear modular equations, **81**, 91
syzygy, 7, 39, **56**
syzygy module, 7, **56**

term, **41**
    divisible, **41**
term-over-position, **43**
TOP order, **44**

uniform shift, 9
unimodular matrix, **21**
unimodularly equivalent, **21**

vanish with multiplicity, *see* root with mult.
vanish with support, *see* root with support
vanishing condition, **90**

weak Popov
    s-weak Popov form, **27**
    shifted weak Popov form, **27**
weak Popov form, **27**
weight, 90
weighted degree, **90**
weighted degree condition, **90**
Welch-Berlekamp decoding, 2
Wu algorithm, 93

zero with multiplicity, *see* root with mult.
zero with support, *see* root with support
Notations:
    $\mathcal{H}.$, *see* assumption
    $\mathcal{O}(\cdot), \tilde{\mathcal{O}}(\cdot)$: asymptotic bounds,
    ω: exponent of matrix multiplication, **4**
    $\mathcal{H}_{\mathsf{MM}(\cdot,\cdot)}$: super-linearity assumption, **156**
    $\mathcal{H}_{\mathsf{M}(\cdot)}$: super-linearity assumption, **155**
    MM': multiplication time, **245**
    MM: mult. time over $\mathbb{K}[X]^{m\times m}$, **155**
    M: multiplication time over $\mathbb{K}[X]$, **155**
    $\overline{\mathsf{MM}''}$: multiplication time, **245**
    $\overline{\mathsf{MM}'}$: multiplication time, **245**
    $\mathbf{c}_j$: coordinate vector, 31, **41**
    $\mathrm{diag}(a_1, \ldots, a_m)$: diagonal matrix,
    $\mathbf{I}_m$: identity $m \times m$ matrix,
    $\mathrm{lm}_{\mathbf{s}}(\mathbf{A})$: **s**-leading matrix, **24**
    $\mathbf{A}_{*,j}$: column j of a matrix,
    $\mathbf{A}_{i,*}$: row i of a matrix,
    $\mathrm{rank}(\mathbf{A})$: rank of a matrix $\mathbf{A}$,
    $\mathbf{A}^{\mathsf{T}}$: transpose of a matrix $\mathbf{A}$,
    $\mathrm{rep}(x, \mathbf{J})$: repetition of x in $\mathbf{J}$, **270**
    $\mathcal{L}$: leading terms of Gröbner basis, 52
    $\mathcal{E} = \{\varepsilon_1, \ldots, \varepsilon_D\}$: monomial basis, **45**
    $\mathcal{B}$: border (of monomial basis), 52
    $\mathcal{S}$: set of exponents,
    $\mathcal{G}$: Gröbner basis,
    $\langle \cdots \rangle$: ideal specified by generators,
    $\mathrm{in}_{\prec}(\mathcal{M})$: $\prec$-initial module, **44**
    $\mathrm{in}_{\prec}(f)$: $\prec$-initial term of f, **44**
    μ: often the multiplicity of a root, **90**
    $\mathrm{nf}_{\prec}(f)$: $\prec$-normal form of f, **46**
    $\prec$: monomial order, **43**
    $\prec_{\mathrm{drl}}$: degree reverse lex. order, **43**
    $\prec_{\mathrm{lex}}$: lexicographic order, **43**
    $\prec^{\mathrm{pot}}$: $\prec$-POT order, **44**
    $\prec^{\mathrm{top}}$: $\prec$-TOP order, **44**

$<^{\mathbf{s}\text{-top}}$: shifted TOP order, 44
$\boldsymbol{\mu}$: multiplicity support of a root, 96
$\mathrm{wdeg}_{w_1,\ldots,w_r}(Q)$: weighted degree of multivariate polynomials, 90
$\mathrm{amp}(\mathbf{s})$: amplitude $\max(\mathbf{s}) - \min(\mathbf{s})$, 85
$\mathrm{cdeg}_{\mathbf{s}}(\mathbf{A})$: $\mathbf{s}$-col. deg. of a matrix, 23
$\mathrm{cdeg}(\mathbf{A})$: column degree of a matrix, 23
$D_{\mathbf{A}}$: generic bound on $\deg(\det(\mathbf{A}))$, 286
$\mathbf{H}$: often a Hermite form, 29
$\mathbf{P}$: often an $\mathbf{s}$-(weak) Popov form, 29
$\mathrm{rdeg}_{\mathbf{s}}(\mathbf{A})$: $\mathbf{s}$-row degree of a matrix, 23
$\mathrm{rdeg}(\mathbf{A})$: row degree of a matrix, 23
$\mathbf{R}$: often an $\mathbf{s}$-reduced form, 24
$\mathbf{X}^{\mathbf{s}}$: shift matrix $\mathrm{diag}(X^{s_1}, \ldots, X^{s_m})$, 24
$\mathbf{s} + c$: add a constant to a tuple, 24
$\mathbf{s}, \mathbf{t}$: shifts (tuples of integers),
$\mathbf{h}$: Hermite shift, 38
$\mathbf{A}_{[i:j,k:l]}$: permuted submatrix of $\mathbf{A}$, 193
$\mathbf{t}_{[i:j]}$: permuted subtuple of $\mathbf{t}$, 193
$|\mathbf{s}|$: sum of the entries of $\mathbf{s} \in \mathbb{Z}^m$,
$\mathbf{0}$: uniform shift $(0, \ldots, 0)$,
$\mathcal{L}^{\mathrm{c}}_{\boldsymbol{\delta}}(\mathbf{A})$: column partial lin. of $\mathbf{A}$, 289
$\mathcal{E}$: expansion-compression mat., 157, 288
$\mathcal{L}^{\mathrm{r}}_{\boldsymbol{\delta}}(\mathbf{A})$: row partial lin. of $\mathbf{A}$, 291
$\boldsymbol{\mathcal{T}}_{\boldsymbol{\delta}}$: partial linearization matrix, 288
$\mathcal{L}^{\mathrm{ab}}_{\mathbf{D},\delta}(\cdot)$: Storjohann's transformation, 177
$\mathcal{E}_{\prec,\boldsymbol{\beta}}(\mathbf{p})/\mathcal{C}_{\prec,\boldsymbol{\beta}}(\mathbf{v})$: expansion/compression of polynomial/scalar vectors, 123
$\mathcal{K}_{\prec,\boldsymbol{\beta}}(\mathbf{M}, \mathbf{F})$: multi-Krylov matrix, 124
$\mathrm{Syz}_{\mathbf{J}}(\mathbf{F})$: module of interpolants, 75
$\mathrm{Syz}_{\mathfrak{M}}(\mathbf{F})$: module of solutions, 81
$\mathrm{Syz}_{\mathbf{D}}(\mathbf{F})$: module of approximants, 70
$\mathrm{Syz}_{\mathbf{M}}(\mathbf{F})$: module of relations, 57
$\phi_{\prec,\boldsymbol{\beta}}$: indexing function, 122
$\mathbf{p} \cdot \mathbf{F}$: module scalar product, 57
$p \cdot \mathbf{f}$: module product, 57
$\mathbb{Z}_{\geqslant 0}$: nonnegative integers,
$\mathbb{Z}$: integers,
$\mathbb{Z}_{>0}$: positive integers,
$\mathrm{Card}(\cdot)$: cardinality,
$\mathbb{K}$: base field,
$\mathbb{L}$: field extension,
$\mathcal{I}$: ideal,
$\mathcal{R}^{m \times n}$: matrices over a ring $\mathcal{R}$,
$\mathcal{M}$: module,
$\mathbb{K}[X]$: polynomials in X over $\mathbb{K}$,
$\mathbb{K}[X]_{\neq 0}$: nonzero polynomials in X,
$\mathbb{K}[\boldsymbol{X}]$ or $\mathbb{K}[X_1, \ldots, X_r]$: multivariate polynomials over $\mathbb{K}$,

Bibliography

[AL94] W. W. Adams and P. Loustaunau. *An introduction to Gröbner bases*. Graduate studies in mathematics. American Mathematical Society, 1994. doi:10.1090/gsm/003.

[Ale02] M. Alekhnovich. Linear Diophantine equations over polynomials and soft decoding of Reed-Solomon codes. In *FOCS'02*, pages 439–448. IEEE, 2002. doi:10.1109/SFCS.2002.1181968.

[Ale05] M. Alekhnovich. Linear Diophantine equations over polynomials and soft decoding of Reed-Solomon codes. *IEEE Trans. Inf. Theory*, 51(7):2257–2265, July 2005. doi:10.1109/TIT.2005.850097.

[BA80] R. R. Bitmead and B. D. O. Anderson. Asymptotically fast solution of Toeplitz and related systems of linear equations. *Linear Algebra Appl.*, 34:103–116, 1980. doi:10.1016/0024-3795(80)90161-5.

[Bak75] G. A. J. Baker. *Essentials of Padé Approximants*. Elsevier Science, 1975.

[BB10] P. Beelen and K. Brander. Key equations for list decoding of Reed-Solomon codes and how to solve them. *J. Symbolic Comput.*, 45(7):773–786, 2010. doi:10.1016/j.jsc.2010.03.010.

[BBF15] J. Berthomieu, B. Boyer, and J.-C. Faugère. Linear algebra for computing gröbner bases of linear recursive multidimensional sequences. In *ISSAC'15*, pages 61–68, New York, NY, USA, 2015. ACM. doi:10.1145/2755996.2756673.

[BBF17] J. Berthomieu, B. Boyer, and J.-C. Faugère. Linear Algebra for Computing Gröbner Bases of Linear Recursive Multidimensional Sequences. *J. Symbolic Comput.*, (in press):?–?, 2017. doi:10.1016/j.jsc.2016.11.005.

[Bec90] B. Beckermann. *Zur Interpolation mit polynomialen Linearkombinationen beliebiger Funktionen*. PhD thesis, Department of Applied Mathematics, University of Hannover, Germany, 1990.

[Bec92] B. Beckermann. A reliable method for computing M-Padé approximants on arbitrary staircases. *J. Comput. Appl. Math.*, 40(1):19–42, 1992. doi:10.1016/0377-0427(92)90039-Z.

[Ber68] E. R. Berlekamp. *Algebraic Coding Theory - Revised edition*. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 2015; first edition 1968.

[Ber11] D. J. Bernstein. Simplified high-speed high-distance list decoding for alternant codes. In *PQCrypto'11*, volume 7071 of *LNCS*, pages 200–216. Springer, 2011. doi:10.1007/978-3-642-25405-5_13.

[BF16] J. Berthomieu and J.-C. Faugère. Guessing linear recurrence relations of sequence tuples and p-recursive sequences with linear algebra. In *ISSAC'16*, pages 95–102, New York, NY, USA, 2016. ACM. doi:10.1145/2930889.2930926.

[BGM96] G. A. Baker and P. R. Graves-Morris. *Padé Approximants*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1996.

[BGY80] R. P. Brent, F. G. Gustavson, and D. Y. Y. Yun. Fast solution of Toeplitz systems of equations and computation of Padé approximants. *Journal of Algorithms*, 1(3):259–295, 1980. doi:10.1016/0196-6774(80)90013-9.

[BH74] J. R. Bunch and J. E. Hopcroft. Triangular factorization and inversion by fast matrix multiplication. *Mathematics of Computation*, 28(125):231–236, 1974. doi:10.2307/2005828.

[BHNW13] P. Beelen, T. Høholdt, J. S. R. Nielsen, and Y. Wu. On rational interpolation-based list-decoding and list-decoding binary goppa codes. *IEEE Trans. Inf. Theory*, 59(6):3269–3281, 2013. doi:10.1109/TIT.2013.2243800.

[BJMS16] A. Bostan, C.-P. Jeannerod, C. Mouilleron, and É. Schost. On matrices with displacement structure: generalized operators and faster algorithms. Manuscript, 2016.

[BJS08] A. Bostan, C.-P. Jeannerod, and É. Schost. Solving structured linear systems with large displacement rank. *Theor. Comput. Sci.*, 407(1-3):155–181, 2008. doi:10.1016/j.tcs.2008.05.014.

[BL94] B. Beckermann and G. Labahn. A uniform approach for the fast computation of matrix-type Padé approximants. *SIAM J. Matrix Anal. Appl.*, 15(3):804–823, July 1994. doi:10.1137/S0895479892230031.

[BL00] B. Beckermann and G. Labahn. Fraction-free computation of matrix rational interpolants and matrix gcds. *SIAM J. Matrix Anal. Appl.*, 22(1):114–144, 2000. doi:10.1137/S0895479897326912.

[BLV99] B. Beckermann, G. Labahn, and G. Villard. Shifted normal forms of polynomial matrices. In *ISSAC'99*, pages 189–196. ACM, 1999. doi:10.1145/309831.309929.

[BLV06] B. Beckermann, G. Labahn, and G. Villard. Normal forms for general polynomial matrices. *J. Symbolic Comput.*, 41(6):708–737, 2006. doi:10.1016/j.jsc.2006.02.001.

[Bon02] Dan Boneh. Finding smooth integers in short intervals using CRT decoding. *J. Comput. Syst. Sci.*, 64(4):768–784, June 2002. doi:10.1006/jcss.2002.1827.

[BPR06] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in Real Algebraic Geometry*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. doi:10.1007/3-540-33099-2.

[Bra10] K. Brander. *Interpolation and List Decoding of Algebraic Codes*. PhD thesis, Technical University of Denmark, 2010.

[BS05] A. Bostan and É. Schost. Polynomial evaluation and interpolation on special sets of points. *J. Complexity*, 21(4):420–446, 2005. doi:10.1016/j.jco.2004.09.009.

[BSS03] A. Bostan, B. Salvy, and É. Schost. Fast algorithms for zero-dimensional polynomial systems using duality. *Appl. Algebra Engrg. Comm. Comput.*, 14(4):239–272, 2003. doi:10.1007/s00200-003-0133-5.

[Buc76] B. Buchberger. A theoretical basis for the reduction of polynomials to canonical forms. *ACM SIGSAM Bull.*, 10(3):19–29, August 1976. doi:10.1145/1088216.1088219.

[Bus08] P. Busse. *Multivariate List Decoding of Evaluation Codes with a Gröbner Basis Perspective*. PhD thesis, University of Kentucky, 2008.

[BvdHP88] Th. G. J. Beelen, G. J. van den Hurk, and C. Praagman. A new method for computing a column reduced polynomial matrix. *Systems and Control Letters*, 10(4):217–224, 1988. doi:10.1016/0167-6911(88)90010-2.

[Cau21] A. Cauchy. *Cours d'analyse de l'École Royale Polytechnique (Analyse algébrique) – Sur la formule de Lagrange relative à l'interpolation*. Imprimerie royale, Paris, 1821.

[CC86] S. Cabay and D.-K. Choi. Algebraic computations of scaled Padé fractions. *SIAM J. Comput.*, 15(1):243–270, 1986. doi:10.1137/0215018.

[CH11] H. Cohn and N. Heninger. Ideal forms of Coppersmith's theorem and Guruswami-Sudan list decoding. In *Innovations in Computer Science*, pages 298–308. Tsinghua University Press, 2011.

[CH12] H. Cohn and N. Heninger. Approximate common divisors via lattices. In *Tenth Algorithmic Number Theory Symposium*, pages 271–293. Mathematical Sciences Publishers (MSP), 2012. doi:10.2140/obs.2013.1.271.

[CH15] H. Cohn and N. Heninger. Ideal forms of Coppersmith's theorem and Guruswami-Sudan list decoding. *Advances in Mathematics of Communications*, 9(3):311–339, 2015. doi:10.3934/amc.2015.9.311.

[Che84] U. Cheng. On the continued fraction and Berlekamp's algorithm (corresp.). *IEEE Trans. Inf. Theory*, 30(3):541–544, May 1984. doi:10.1109/TIT.1984.1056906.

[CJN+15] M. Chowdhury, C.-P. Jeannerod, V. Neiger, É. Schost, and G. Villard. Faster algorithms for multivariate interpolation with multiplicities and simultaneous polynomial approximations. *IEEE Trans. Inf. Theory*, 61(5):2370–2387, 2015. doi:10.1109/TIT.2015.2416068.

[CK91] D. G. Cantor and E. Kaltofen. On fast multiplication of polynomials over arbitrary algebras. *Acta Inform.*, 28(7):693–701, 1991. doi:10.1007/BF01178683.

[Cla75] G. Claessens. A new look at the Padé table and the different methods for computing its elements. *J. Comput. Appl. Math.*, 1(3):141–152, 1975. doi:10.1016/0771-050X(75)90032-7.

[CLO05] D. Cox, J. Little, and D. O'Shea. *Using Algebraic Geometry (second edition)*. Springer-Verlag New-York, 2005. doi:10.1007/b138611.

[CLO07] D. A. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms (third edition)*. Springer-Verlag New-York, 2007. doi:10.1007/978-0-387-35651-8.

[Coa66] J. Coates. On the algebraic approximation of functions, I–III. *Indag. Math.*, 69:421–461, 1966. doi:10.1016/S1385-7258(66)50049-X.

[Coa67] J. Coates. On the algebraic approximation of functions, IV. *Indag. Math.*, 70:205–212, 1967. doi:10.1016/S1385-7258(67)50033-1.

[Cop96] Don Coppersmith. Finding a small root of a univariate modular equation. In Ueli Maurer, editor, *Advances in Cryptology — EUROCRYPT'96*, volume 1070 of *Lecture Notes in Computer Science*, pages 155–165. Springer Berlin / Heidelberg, 1996. doi:10.1007/3-540-68339-9_14.

[CW90] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *J. Symbolic Comput.*, 9(3):251–280, 1990. doi:10.1016/S0747-7171(08)80013-2.

[Dan37] A. M. Danilevskii. The numerical solution of the secular equation. *Matem. Sbornik*, 44(2):169–171, 1937. In Russian.

[DF04] D. S. Dummit and R. M. Foote. *Abstract Algebra*. John Wiley & Sons, 2004.

[DGH12] C. Devet, I. Goldberg, and N. Heninger. Optimally robust private information retrieval. In *USENIX Security 12*, pages 269–283. USENIX, 2012.

[DL78] R. A. DeMillo and R. J. Lipton. A probabilistic remark on algebraic program testing. *Inform. Process. Lett.*, 7(4):193–195, 1978. doi:10.1016/0020-0190(78)90067-4.

[Dor87] J. L. Dornstetter. On the equivalence between Berlekamp's and Euclid's algorithms. *IEEE Trans. Inf. Theory*, 33(3):428–431, May 1987. doi:10.1109/TIT.1987.1057299.

[EGV00] W. Eberly, M. Giesbrecht, and G. Villard. On computing the determinant and smith form of an integer matrix. In *FOCS'00*, pages 675–687, Washington, DC, USA, 2000. IEEE Computer Society. doi:10.1109/SFCS.2000.892335.

[Eis95] D. Eisenbud. *Commutative Algebra: with a View Toward Algebraic Geometry*. Graduate Texts in Mathematics. Springer, New York, Berlin, Heildelberg, 1995. doi:10.1007/978-1-4612-5350-1.

[Eis05] D. Eisenbud. *The Geometry of Syzygies*. Graduate Texts in Mathematics. Springer, New York, Berlin, Heildelberg, 2005. doi:10.1007/b137572.

[FF92] P. Fitzpatrick and J. Flynn. A Gröbner basis technique for Padé approximation. *J. Symbolic Comput.*, 13(2):133–138, 1992. doi:10.1016/S0747-7171(08)80087-9.

[FG06] J. B. Farr and S. Gao. Gröbner bases and generalized Padé approximation. *Mathematics of Computation*, 74(253):461–473, 1 2006. doi:10.1090/S0025-5718-05-01790-4.

[FGHR13] J.-C. Faugère, P. Gaudry, L. Huot, and G. Renault. Polynomial systems solving by fast linear algebra. *CoRR*, abs/1304.6039, 2013. URL: http://arxiv.org/abs/1304.6039.

[FGHR14] J.-C. Faugère, P. Gaudry, L. Huot, and G. Renault. Sub-cubic change of ordering for Gröbner basis: a probabilistic approach. In *ISSAC'14*, pages 170–177, New York, NY, USA, 2014. ACM. doi:10.1145/2608628.2608669.

[FGLM93] J.-C. Faugère, P. Gianni, D. Lazard, and T. Mora. Efficient computation of zero-dimensional Gröbner bases by change of ordering. *Journal of Symbolic Computation*, 16(4):329–344, 1993. doi:10.1006/jsco.1993.1051.

[Fit95] P. Fitzpatrick. On the key equation. *IEEE Trans. Inf. Theory*, 41(5):1290–1302, 1995. doi:10.1109/18.412677.

[Fit97] P. Fitzpatrick. Solving a Multivariable Congruence by Change of Term Order. *J. Symbolic Comput.*, 24(5):575–589, 1997. doi:10.1006/jsco.1997.0153.

[FM11] J.-C. Faugère and C. Mou. Fast algorithm for change of ordering of zero-dimensional gröbner bases with sparse multiplication matrices. In *ISSAC'11*, pages 115–122, New York, NY, USA, 2011. ACM. doi:10.1145/1993886.1993908.

[FM17] J.-C. Faugère and C. Mou. Sparse FGLM algorithms. *J. Symbolic Comput.*, 80, Part 3:538–569, 2017. doi:10.1016/j.jsc.2016.07.025.

[For75] G. D. Forney, Jr. Minimal Bases of Rational Vector Spaces, with Applications to Multivariable Linear Systems. *SIAM Journal on Control*, 13(3):493–520, 1975. doi:10.1137/0313029.

[FT91] G. L. Feng and K. K. Tzeng. A generalization of the Berlekamp-Massey algorithm for multisequence shift-register synthesis with applications to decoding cyclic codes. *IEEE Trans. Inf. Theory*, 37(5):1274–1287, 1991. doi:10.1109/18.133246.

[Ged73] K. O. Geddes. *Algorithms for Analytic Approximation (to a Formal Power-series).* PhD thesis, University of Toronto, Canada, 1973.

[Ged79] K. O. Geddes. Symbolic computation of Padé approximants. *ACM Trans. Math. Softw.*, 5(2):218–233, June 1979. doi:10.1145/355826.355835.

[GG13] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra (third edition).* Cambridge University Press, 2013. doi:10.1017/CBO9781139856065.

[GJV03] P. Giorgi, C.-P. Jeannerod, and G. Villard. On the complexity of polynomial matrix computations. In *ISSAC'03*, pages 135–142. ACM, 2003. doi:10.1145/860854.860889.

[GL14] P. Giorgi and R. Lebreton. Online order basis algorithm and its impact on the block Wiedemann algorithm. In *ISSAC'14*, pages 202–209, New York, NY, USA, 2014. ACM. doi:10.1145/2608628.2608647.

[GR06] P. Gaborit and O. Ruatta. Improved Hermite multivariate polynomial interpolation. In *ISIT'06*, pages 143–147. IEEE, 2006. doi:10.1109/ISIT.2006.261691.

[GR08] V. Guruswami and A. Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Trans. Inf. Theory*, 54(1):135–150, 2008. doi:10.1109/TIT.2007.911222.

[GS98] V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and algebraic-geometric codes. In *FOCS'98*, pages 28–39, November 1998. doi:10.1109/SFCS.1998.743426.

[GS99] V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and algebraic-geometry codes. *IEEE Trans. Inf. Theory*, 45(6):1757–1767, 1999. doi:10.1109/18.782097.

[GS11] S. Gupta and A. Storjohann. Computing Hermite forms of polynomial matrices. In *ISSAC'11*, pages 155–162. ACM, 2011. doi:10.1145/1993886.1993913.

[GSSV12] S. Gupta, S. Sarkar, A. Storjohann, and J. Valeriote. Triangular x-basis decompositions and derandomization of linear algebra algorithms over $K[x]$. *J. Symbolic Comput.*, 47(4):422–453, 2012. doi:10.1016/j.jsc.2011.09.006.

[GTV90] M. Girault, P. Toffin, and B. Vallée. Computation of approximate L-th roots modulo n and application to cryptography. In *CRYPTO '88*, pages 100–117, New York, NY, USA, 1990. Springer-Verlag New York, Inc. doi:10.1007/0-387-34799-2_9.

[Gup11] S. Gupta. Hermite forms of polynomial matrices. Master's thesis, University of Waterloo, Canada, 2011.

[GY79] F. G. Gustavson and D. Y. Y. Yun. Fast algorithms for rational Hermite approximation and solution of Toeplitz systems. *IEEE Trans. Circuits Syst.*, 26(9):750–755, 1979. doi:10.1109/TCS.1979.1084696.

[Has36] H. Hasse. Theorie der höheren Differentiale in einem algebraischen Funktionenkörper mit vollkommenem Konstantenkörper bei beliebiger Charakteristik. *J. Reine Angew. Math.*, 175:50–54, 1936. doi:10.1515/crll.1936.175.50.

[Hås86] Johan Håstad. On using RSA with low exponent in a public key network. In *Lecture Notes in Computer Sciences; 218 on Advances in Cryptology—CRYPTO'85*, pages 403–408, New York, NY, USA, 1986. Springer-Verlag New York, Inc. doi:10.1007/3-540-39799-X_29.

[Her51] C. Hermite. Sur l'introduction des variables continues dans la théorie des nombres. *Journal für die reine und angewandte Mathematik*, 41:191–216, 1851. doi:10.1515/crll.1851.41.191.

[Her93] C. Hermite. Sur la généralisation des fractions continues algébriques. *Annali di Matematica Pura ed Applicata (1867-1897)*, 21(1):289–308, 1893. doi:10.1007/BF02420446.

[HG01] N. Howgrave-Graham. Approximate integer common divisors. In *CaLC'01*, pages 51–66, London, UK, 2001. Springer-Verlag. doi:10.1007/3-540-44670-2_6.

[HM91] J. L. Hafner and K. McCurley. Asymptotically fast triangularization of matrices over rings. *SIAM Journal on Computing*, 20(6):1068–1083, 1991. doi:10.1137/0220067.

[Ili89] C. S. Iliopoulos. Worst-case complexity bounds on algorithms for computing the canonical structure of finite abelian groups and the Hermite and Smith normal forms of an integer matrix. *SIAM J. Comput.*, 18(4):658–669, August 1989. doi:10.1137/0218045.

[IMH82] O. H. Ibarra, S. Moran, and R. Hui. A generalization of the fast LUP matrix decomposition algorithm and applications. *Journal of Algorithms*, 3(1):45–56, 1982. doi:10.1016/0196-6774(82)90007-4.

[Jag64] H. Jager. A multidimensional generalization of the Padé table, I–VI. *Indag. Math.*, 67:193–249, 1964. doi:10.1016/S1385-7258(64)50023-2.

[JNSV16] C.-P. Jeannerod, V. Neiger, É. Schost, and G. Villard. Fast computation of minimal interpolation bases in Popov form for arbitrary shifts. In *ISSAC'16*, pages 295–302. ACM, 2016. doi:10.1145/2930889.2930928.

[JNSV17] C.-P. Jeannerod, V. Neiger, É. Schost, and G. Villard. Computing minimal interpolation bases, 2017. doi:10.1016/j.jsc.2016.11.015.

[JV05] C.-P. Jeannerod and G. Villard. Essentially optimal computation of the inverse of generic polynomial matrices. *J. Complexity*, 21(1):72–86, 2005. doi:10.1016/j.jco.2004.03.005.

[Kai80] T. Kailath. *Linear Systems*. Prentice-Hall, 1980.

[Kal94] E. Kaltofen. Asymptotically fast solution of Toeplitz-like singular linear systems. In *ISSAC'94*, pages 297–304. ACM, 1994. doi:10.1145/190347.190431.

[KG85] W. Keller-Gehrig. Fast algorithms for the characteristic polynomial. *Theoretical Computer Science*, 36:309–317, 1985. doi:10.1016/0304-3975(85)90049-0.

[KKS90] E. Kaltofen, M.S. Krishnamoorthy, and D. Saunders. Parallel algorithms for matrix normal forms. *Linear Algebra Appl.*, 136:189–208, 1990. doi:10.1016/0024-3795(90)90028-B.

[KMV11] R. Kötter, J. Ma, and A. Vardy. The re-encoding transformation in algebraic list-decoding of Reed-Solomon codes. *IEEE Trans. Inf. Theory*, 57(2):633–647, 2011. doi:10.1109/TIT.2010.2096034.

[Knu70] D. E. Knuth. The analysis of algorithms. In *Congrès int. Math., Nice, France*, volume 3, pages 269–274, 1970.

[Köt96] R. Kötter. Fast generalized minimum-distance decoding of algebraic-geometry and Reed-Solomon codes. *IEEE Trans. Inf. Theory*, 42(3):721–737, 1996. doi:10.1109/18.490540.

[KR00] M. Kreuzer and L. Robbiano. *Computational Commutative Algebra 1*. Springer Berlin Heidelberg, 2000. doi:10.1007/978-3-540-70628-1.

[Kry31] A. N. Krylov. On the numerical solution of the equation by which, in technical questions, frequencies of small oscillations of material systems are determined. *Izvestiya Akademii Nauk SSSR*, 7(4):491–539, 1931. In Russian. URL: http://mi.mathnet.ru/izv5215.

[KS91] E. Kaltofen and D. Saunders. On Wiedemann's method of solving sparse linear systems. In *AAECC-9*, volume 539 of *LNCS*, pages 29–38. Springer, 1991. doi:10.1007/3-540-54522-0_93.

[KU11] K. S. Kedlaya and C. Umans. Fast polynomial factorization and modular composition. *SIAM J. Comput.*, 40(6):1767–1802, 2011. doi:10.1137/08073408X.

[KV03a] R. Kötter and A. Vardy. Algebraic soft-decision decoding of Reed-Solomon codes. *IEEE Trans. Inf. Theory*, 49(11):2809–2825, 2003. doi:10.1109/TIT.2003.819332.

[KV03b] R. Kötter and A. Vardy. A complexity reducing transformation in algebraic list decoding of Reed-Solomon codes. In *ITW2003*, pages 10–13. IEEE, 2003. doi:10.1109/ITW.2003.1216682.

[Lec01] G. Lecerf. Private communication to É. Schost, 2001.

[LG14] F. Le Gall. Powers of tensors and fast matrix multiplication. In *ISSAC'14*, pages 296–303. ACM, 2014. doi:10.1145/2608628.2608664.

[LNZ16] G. Labahn, V. Neiger, and W. Zhou. Fast, deterministic computation of the Hermite normal form and determinant of a polynomial matrix, 2016. URL: http://arxiv.org/abs/1607.04176.

[LO06] K. Lee and M. E. O'Sullivan. An interpolation algorithm using Gröbner bases for soft-decision decoding of Reed-Solomon codes. In *ISIT'06*, pages 2032–2036, July 2006. doi:10.1109/ISIT.2006.261906.

[LO08] K. Lee and M. E. O'Sullivan. List decoding of Reed-Solomon codes from a Gröbner basis perspective. *J. Symbolic Comput.*, 43(9):645–658, 2008. doi:10.1016/j.jsc.2008.01.002.

[Lüb83] W. Lübbe. *Über ein allgemeines Interpolationsproblem — lineare Identitäten zwischen benachbarten Lösungssystemen*. PhD thesis, Department of Applied Mathematics, University of Hannover, Germany, 1983.

[Mac02] F. S. Macaulay. Some formulae in elimination. *Proceedings of the London Mathematical Society*, s1-35(1):3–27, 1902. doi:10.1112/plms/s1-35.1.3.

[Mac16] F. S. Macaulay. *The Algebraic Theory of Modular Systems*. Cambridge Tracts in Mathematics and Mathematical Physics. Cambridge University Press, 1916.

[Mah32] K. Mahler. Zur approximation der Exponentialfunktion und des Logarithmus. I–II. *Journal für die reine und angewandte Mathematik*, 166:118–150, 1932. doi:10.1515/crll.1932.166.118.

[Mah53] K. Mahler. On the approximation of logarithms of algebraic numbers. *Philosophical Transactions of the Royal Society of London, Series A*, 245(898):371–398, 1953. doi:10.1098/rsta.1953.0001.

[Mah68] K. Mahler. Perfect systems. *Composit. Math.*, 19(2):95–166, 1968.

[Mas69] James L. Massey. Shift-register synthesis and BCH decoding. *IEEE Trans. Inf. Theory*, 15(1):122–127, January 1969. doi:10.1109/TIT.1969.1054260.

[MB82] H. M. Möller and B. Buchberger. The construction of multivariate polynomials with preassigned zeros. In *EUROCAM'82*, volume 144 of *LNCS*, pages 24–31. Springer, 1982. doi:10.1007/3-540-11607-9_3.

[McE03] R. J. McEliece. The Guruswami-Sudan decoding algorithm for Reed-Solomon codes, 2003. IPN Progress Report 42-153.

[Mid11] J. Middeke. *A computational view on normal forms of matrices of Ore polynomials*. Risc technical report 11-10, Research Institute for Symbolic Computation (RISC), July 2011.

[Mil75] W. H. Mills. Continued fractions and linear recurrences. *Mathematics of Computation*, 29(129):173–180, January 1975. doi:10.1090/S0025-5718-1975-0369276-7.

[MMM93] M. G. Marinari, H. M. Möller, and T. Mora. Gröbner bases of ideals defined by functionals with an application to ideals of projective points. *Appl. Algebra Engrg. Comm. Comput.*, 4(2):103–145, 1993. doi:10.1007/BF01386834.

[Moe73] R. T. Moenck. Fast computation of GCDs. In *Proc. 5th ACM Symp. Theory Comp.*, pages 142–151, 1973. doi:10.1145/800125.804045.

[Mor80] M. Morf. Doubling algorithms for Toeplitz and related equations. In *IEEE Conference on Acoustics, Speech, and Signal Processing*, pages 954–959. IEEE, 1980. doi:10.1109/ICASSP.1980.1171074.

[Mor09] T. Mora. The FGLM problem and Möller's algorithm on zero-dimensional ideals. In M. Sala, S. Sakata, T. Mora, C. Traverso, and L. Perret, editors, *Gröbner Bases, Coding, and Cryptography*, pages 27–45, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. doi:10.1007/978-3-540-93806-4_3.

[MS78] R. J. McEliece and J. B. Shearer. A property of Euclid's algorithm and an application to Padé approximation. *SIAM J. Appl. Math.*, 34(4):611–615, 1978. doi:10.1137/0134048.

[MS91] G. Moreno-Socias. *Autour de la fonction de Hilbert-Samuel (escaliers d'idéaux polynomiaux)*. PhD thesis, École Polytechnique, France, 1991. URL: http://www.theses.fr/1991EPXX0034.

[MS03a] G. Moreno-Socias. Degrevlex Gröbner bases of generic complete intersections. *Journal of Pure and Applied Algebra*, 180(3):263–283, 2003. doi:10.1016/S0022-4049(02)00297-9.

[MS03b] T. Mulders and A. Storjohann. On lattice reduction for polynomial matrices. *J. Symbolic Comput.*, 35:377–401, 2003. doi:10.1016/S0747-7171(02)00139-6.

[MS05] E. Miller and B. Sturmfels. *Combinatorial Commutative Algebra*. Graduate texts in mathematics. Springer, New York, 2005. doi:10.1007/b138602.

[Nei16] V. Neiger. Fast computation of shifted Popov forms of polynomial matrices via systems of modular polynomial equations. In *ISSAC'16*, pages 365–372. ACM, 2016. doi:10.1145/2930889.2930936.

[NH00] R. R. Nielsen and T. Høholdt. Decoding Reed-Solomon codes beyond half the minimum distance. In *Coding Theory, Cryptography and Related Areas*, pages 221–236. Springer, 2000. doi:10.1007/978-3-642-57189-3_20.

[Nie13] J. S. R. Nielsen. *List Decoding of Algebraic Codes*. PhD thesis, Technical University of Denmark, 2013.

[Nie14] J. S. R. Nielsen. Fast Kötter-Nielsen-Høholdt interpolation in the Guruswami-Sudan algorithm. In *ACCT'14*, 2014. URL: http://arxiv.org/abs/1406.0053.

[Nus80] H. Nussbaumer. Fast polynomial transform algorithms for digital convolution. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(2):205–215, 1980. doi:10.1109/TASSP.1980.1163372.

[NZ04] M. Nüsken and M. Ziegler. Fast multipoint evaluation of bivariate polynomials. In *Algorithms – ESA 2004*, pages 544–555. Springer Berlin Heidelberg, 2004. doi:10.1007/978-3-540-30140-0_49.

[OF02] H. O'Keeffe and P. Fitzpatrick. Gröbner basis solutions of constrained interpolation problems. *Linear Algebra Appl.*, 351:533–551, 2002. doi:10.1016/S0024-3795(01)00509-2.

[OF07] H. O'Keeffe and P. Fitzpatrick. Gröbner basis approach to list decoding of algebraic geometry codes. *Appl. Algebra Engrg. Comm. Comput.*, 18(5):445–466, 2007. doi:10.1007/s00200-007-0048-7.

[OS99] V. Olshevsky and M. A. Shokrollahi. A displacement approach to efficient decoding of algebraic-geometric codes. In *STOC'99*, pages 235–244. ACM, 1999. doi:10.1145/301250.301311.

[Pad94] H. Padé. Sur la généralisation des fractions continues algébriques. *Journal de Mathématiques Pures et Appliquées*, pages 291–330, 1894.

[Pan01] V. Y. Pan. *Structured Matrices and Polynomials*. Birkhäuser/Springer, Boston/New York, 2001. doi:10.1007/978-1-4612-0129-8.

[Pap94] C. H. Papadimitriou. *Computational Complexity*. Theoretical Computer Science. Addison-Wesley, 1994.

[Pas87] S. Paszkowski. Recurrence relations in Padé-Hermite approximation. *J. Comput. Appl. Math.*, 19(1):99–107, July 1987. doi:10.1016/0377-0427(87)90177-4.

[Pop72] V. M. Popov. Invariant description of linear, time-invariant controllable systems. *SIAM Journal on Control*, 10(2):252–264, 1972. doi:10.1137/0310020.

[PV05] F. Parvaresh and A. Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *FOCS'05*, pages 285–294. IEEE, 2005. doi:10.1109/SFCS.2005.29.

[Rei03] J.-R. Reinhard. Algorithme LLL polynomial et applications. Master's thesis, École Polytechnique, Paris, France, 2003. URL: https://hal.inria.fr/hal-01101550.

[Rob86] L. Robbiano. On the theory of graded structures. *J. Symbolic Comput.*, 2(2):139–170, 1986. doi:10.1016/S0747-7171(86)80019-0.

[Rot07] R. M. Roth. *Introduction to Coding Theory*. Cambridge University Press, 2007. doi:10.1017/CBO9780511808968.

[Rou99] F. Rouillier. Solving zero-dimensional systems through the rational univariate representation. *Appl. Algebra Engrg. Comm. Comput.*, 9(5):433–461, 1999. doi:10.1007/s002000050114.

[RR00] R. M. Roth and G. Ruckenstein. Efficient decoding of Reed-Solomon codes beyond half the minimum distance. *IEEE Trans. Inf. Theory*, 46(1):246–257, 2000. doi:10.1109/18.817522.

[RS16] J. Rosenkilde and A. Storjohann. Algorithms for simultaneous Padé approximations. In *ISSAC'16*, pages 405–412, New York, NY, USA, 2016. ACM. doi:10.1145/2930889.2930933.

[Sak88] 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. doi:10.1016/S0747-7171(88)80033-6.

[Sak90] S. Sakata. Extension of the berlekamp-massey algorithm to n dimensions. *Inform. and Comput.*, 84(2):207–239, 1990. doi:10.1016/0890-5401(90)90039-K.

[Sch71] A. Schönhage. Schnelle Berechnung von Kettenbruchentwicklungen. *Acta Inform.*, 1:139–144, 1971. In German. doi:10.1007/BF00289520.

[Sch77] A. Schönhage. Fast multiplication of polynomials over fields of characteristic 2. *Acta Inform.*, 7(4):395–398, 1977. doi:10.1007/BF00289470.

[Sch80] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980. doi:10.1145/322217.322225.

[Ser87] A. V. Sergeyev. A recursive algorithm for Padé-Hermite approximations. *USSR Comput. Math. Math. Phys.*, 26(2):17–22, July 1987. doi:10.1016/0041-5553(86)90003-0.

[Sho91] V. Shoup. A fast deterministic algorithm for factoring polynomials over finite fields of small characteristic. In *ISSAC'91*, pages 14–21. ACM, 1991. doi:10.1145/120694.120697.

[SKHN75] Y. Sugiyama, M. Kasahara, S. Hirasawa, and T. Namekawa. A method for solving key equation for decoding Goppa codes. *Information and Control*, 27(1):87–99, 1975. doi:10.1016/S0019-9958(75)90090-X.

[SL96] A. Storjohann and G. Labahn. Asymptotically fast computation of Hermite normal forms of integer matrices. In *ISSAC'96*, pages 259–266. ACM, 1996. doi:10.1145/236869.237083.

[SS11] S. Sarkar and A. Storjohann. Normalization of row reduced matrices. In *ISSAC'11*, pages 297–304. ACM, 2011. doi:10.1145/1993886.1993931.

[Sto00] A. Storjohann. *Algorithms for Matrix Canonical Forms*. PhD thesis, Swiss Federal Institute of Technology – ETH, 2000.

[Sto03] A. Storjohann. High-order lifting and integrality certification. *J. Symbolic Comput.*, 36(3-4):613–648, 2003. doi:10.1016/S0747-7171(03)00097-X.

[Sto06] A. Storjohann. Notes on computing minimal approximant bases. In *Challenges in Symbolic Computation Software*, Dagstuhl Seminar Proceedings, 2006. URL: http://drops.dagstuhl.de/opus/volltexte/2006/776.

[Sud97] M. Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *J. Complexity*, 13(1):180–193, 1997. doi:10.1006/jcom.1997.0439.

[SV05] A. Storjohann and G. Villard. Computing the rank and a small nullspace basis of a polynomial matrix. In *ISSAC'05*, pages 309–316, New York, NY, USA, 2005. ACM. doi:10.1145/1073884.1073927.

[Syl53] J. J. Sylvester. On a Theory of the Syzygetic Relations of Two Rational Integral Functions, Comprising an Application to the Theory of Sturm's Functions, and That of the Greatest Algebraical Common Measure. *Philosophical Transactions of the Royal Society of London*, 143:407–548, 1853. doi:10.1098/rstl.1853.0018.

[Tho01] E. Thomé. Fast computation of linear generators for matrix sequences and application to the block Wiedemann algorithm. In *ISSAC'01*, pages 323–331, New York, NY, USA, 2001. ACM. doi:10.1145/384101.384145.

[Tho02] E. Thomé. Subquadratic computation of vector generating polynomials and improvement of the block Wiedemann algorithm. *J. Symbolic Comput.*, 33(5):757–775, 2002. doi:10.1006/jsco.2002.0533.

[Tri10] P. V. Trifonov. Efficient interpolation in the Guruswami-Sudan algorithm. *IEEE Trans. Inf. Theory*, 56(9):4341–4349, 2010. doi:10.1109/TIT.2010.2053901.

[VBB91] M. Van Barel and A. Bultheel. The computation of non-perfect Padé-Hermite approximants. *Numer. Algorithms*, 1(3):285–304, 1991. doi:10.1007/BF02142327.

[VBB92] M. Van Barel and A. Bultheel. A general module theoretic framework for vector M-Padé and matrix rational interpolation. *Numer. Algorithms*, 3:451–462, 1992. doi:10.1007/BF02141952.

[Vil96] G. Villard. Computing Popov and Hermite forms of polynomial matrices. In *ISSAC'96*, pages 250–258. ACM, 1996. doi:10.1145/236869.237082.

[War74] D. D. Warner. *Hermite interpolation with rational functions*. PhD thesis, University of California, San Diego, 1974.

[War76] D. D. Warner. Kronecker's algorithm for Hermite interpolation with an application to sphere drag in a fluid-filled tube. In *Proceedings of a Workshop on Padé Approximation, eds. D. Bessis, J. Gilewicz and P. Mery, CNRS Marseille*, pages 48–71, 1976.

[WB86] L. R. Welch and E. R. Berlekamp. Error correction for algebraic block codes, December 30 1986. US Patent 4,633,470.

[Wol74] W. A. Wolovich. *Linear Multivariable Systems*, volume 11 of *Applied Mathematical Sciences*. Springer-Verlag New-York, 1974. doi:10.1007/978-1-4612-6392-0.

[WS79] L. Welch and R. Scholtz. Continued fractions and Berlekamp's algorithm. *IEEE Trans. Inf. Theory*, 25(1):19–27, Jan 1979. doi:10.1109/TIT.1979.1055987.

[Wu08] Y. Wu. New list decoding algorithms for Reed-Solomon and BCH codes. *IEEE Trans. Inf. Theory*, 54(8):3611 –3630, August 2008. doi:10.1109/TIT.2008.926355.

[Zeh13] A. Zeh. *Algebraic Soft- and Hard-Decision Decoding of Generalized Reed–Solomon and Cyclic Codes*. PhD thesis, École Polytechnique, 2013.

[ZGA11] A. Zeh, C. Gentner, and D. Augot. An interpolation procedure for list decoding Reed-Solomon codes based on generalized key equations. *IEEE Trans. Inf. Theory*, 57(9):5946–5959, 2011. doi:10.1109/TIT.2011.2162160.

[Zho12] W. Zhou. *Fast Order Basis and Kernel Basis Computation and Related Problems*. PhD thesis, University of Waterloo, 2012.

[Zie68] N. Zierler. Linear recurring sequences and error-correcting codes. In H. B. Mann, editor, *Error Correcting Codes (Proc. Sympos. Math. Res. Center, Madison, Wis., 1968)*, pages 47–59. Wiley, 1968.

[Zip79] R. Zippel. Probabilistic algorithms for sparse polynomials. In *EUROSAM'79*, volume 72 of *LNCS*, pages 216–226. Springer, 1979.

[ZL12] W. Zhou and G. Labahn. Efficient algorithms for order basis computation. *J. Symbolic Comput.*, 47(7):793–819, 2012. doi:10.1016/j.jsc.2011.12.009.

[ZL13] W. Zhou and G. Labahn. Computing column bases of polynomial matrices. In *ISSAC'13*, pages 379–386, New York, NY, USA, 2013. ACM. doi:10.1145/2465506.2465947.

[ZL14a] W. Zhou and G. Labahn. Fast and deterministic computation of the determinant of a polynomial matrix, 2014. URL: http://arxiv.org/abs/1409.5462.

[ZL14b] W. Zhou and G. Labahn. Unimodular completion of polynomial matrices. In *ISSAC'14*, pages 413–420, New York, NY, USA, 2014. ACM. doi:10.1145/2608628.2608640.

[ZL16] W. Zhou and G. Labahn. A fast, deterministic algorithm for computing a Hermite normal form of a polynomial matrix, 2016. URL: http://arxiv.org/abs/1602.02049.

[ZLS12] W. Zhou, G. Labahn, and A. Storjohann. Computing minimal nullspace bases. In *ISSAC'12*, pages 366–373. ACM, 2012. doi:10.1145/2442829.2442881.

[ZLS15] W. Zhou, G. Labahn, and A. Storjohann. A deterministic algorithm for inverting a polynomial matrix. *J. Complexity*, 31(2):162–173, 2015. doi:10.1016/j.jco.2014.09.004.

Résumé

Dans cette thèse, nous étudions des algorithmes pour un problème de recherche de relations à une ou plusieurs variables. Il généralise celui de calculer une solution à un système d'équations linéaires modulaires sur un anneau de polynômes, et inclut par exemple le calcul d'approximants de Hermite-Padé ou d'interpolants bivariés. Plutôt qu'une seule solution, nous nous attacherons à calculer un ensemble de générateurs possédant de bonnes propriétés.

Précisément, l'entrée de notre problème consiste en un module de dimension finie spécifié par l'action des variables sur ses éléments, et en un certain nombre d'éléments de ce module ; il s'agit de calculer une base de Gröbner du module des relations entre ces éléments. En termes d'algèbre linéaire, l'entrée décrit une matrice avec une structure de type Krylov, et il s'agit de calculer sous forme compacte une base du noyau de cette matrice.

Nous proposons plusieurs algorithmes en fonction de la forme des matrices de multiplication qui représentent l'action des variables. Dans le cas d'une matrice de Jordan, nous accélérons le calcul d'interpolants multivariés sous certaines contraintes de degré ; nos résultats pour une forme de Frobenius permettent d'accélérer le calcul de formes normales de matrices polynomiales univariées. Enfin, dans le cas de plusieurs matrices denses, nous accélérons le changement d'ordre pour des bases de Gröbner d'idéaux multivariés zéro-dimensionnels.

Abstract

In this thesis, we study algorithms for a problem of finding relations in one or several variables. It generalizes that of computing a solution to a system of linear modular equations over a polynomial ring, including in particular the computation of Hermite-Padé approximants and bivariate interpolants. Rather than a single solution, we aim at computing generators of the solution set which have good properties.

Precisely, the input of our problem consists of a finite-dimensional module given by the action of the variables on its elements, and of some elements of this module; the goal is to compute a Gröbner basis of the module of syzygies between these elements. In terms of linear algebra, the input describes a matrix with a type of Krylov structure, and the goal is to compute a compact representation of a basis of the nullspace of this matrix.

We propose several algorithms in accordance with the structure of the multiplication matrices which specify the action of the variables. In the case of a Jordan matrix, we accelerate the computation of multivariate interpolants under degree constraints; our result for a Frobenius matrix leads to a faster algorithm for computing normal forms of univariate polynomial matrices. In the case of several dense matrices, we accelerate the change of monomial order for Gröbner bases of multivariate zero-dimensional ideals.