Algorithms for Normal Forms for Matrices of Polynomials and Ore Polynomials

by

Howard Cheng

A thesis
presented to the University of Waterloo
in fulfilment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2003

Abstract

In this thesis we study algorithms for computing normal forms for matrices of Ore polynomials while controlling coefficient growth. By formulating row reduction as a linear algebra problem, we obtain a fraction-free algorithm for row reduction for matrices of Ore polynomials. The algorithm allows us to compute the rank and a basis of the left nullspace of the input matrix. When the input is restricted to matrices of shift polynomials and ordinary polynomials, we obtain fraction-free algorithms for computing row-reduced forms and weak Popov forms. These algorithms can be used to compute a greatest common right divisor and a least common left multiple of such matrices. Our fraction-free row reduction algorithm can be viewed as a generalization of subresultant algorithms. The linear algebra formulation allows us to obtain bounds on the size of the intermediate results and to analyze the complexity of our algorithms.

We then make use of the fraction-free algorithm as a basis to formulate modular algorithms for computing a row-reduced form, a weak Popov form, and the Popov form of a polynomial matrix. By examining the linear algebra formulation, we develop criteria for detecting unlucky homomorphisms and determining the number of homomorphic images required.

Acknowledgements

I wish to thank my supervisor, George Labahn, for his support for the duration of this work. Without his suggestions, criticisms, friendship, and financial support, it would not have been possible to complete this work. I also learned much from Bernhard Beckermann when we worked together to obtain some of the results reported in this thesis. I also wish to thank Dr. Keith Geddes, Dr. Mark Giesbrecht, Dr. David Saunders, and Dr. Cameron Stewart for serving on the thesis committee.

Past and present members of the Symbolic Computation Group and Ontario Research Centre for Computer Algebra provided a wonderful environment for conducting research. I had many interesting discussions with Reinhold Burger, Claude-Pierre Jeannerod, Ha Le, Wen-shin Lee, Ziming Li, Arne Storjohann, and Eugene Zima. In particular, I wish to thank Reinhold Burger for putting up with me as an officemate during my stay at Waterloo. Additionally, a number of people have made my life at Waterloo much more interesting and enjoyable. Amélie Bélanger, Reinhold Burger, Yann d'Halluin, David and Elizabeth Pooley, and Dana Wilkinson have all contributed to many activities in my social life, including sports, bridge, cooking, and eating.

I would like to thank my parents Ben and Clara and my brother Vince for their support. I also wish to thank Katerina Carastathis for her friendship over the years, and for being a wonderful person to talk to when I needed to.

"Algebraic symbols are used when you do not know what you are talking about."

Contents

Chapter 1

Introduction

Normal forms are tools commonly used in the study of equivalence of mathematical objects. Such forms are associated with transformation functions which preserve the equivalence. Objects in normal form are unchanged by such a transformation [36]. In effect, objects in normal forms are representatives chosen from a class of equivalent objects. Certain properties of an object can be obtained easily once it has been transformed into normal form. Normal forms also have many applications in computer algebra, including simplification, equivalence tests, and determination of properties from mathematical objects (sometimes called invariants).

There are many examples of normal forms in the area of linear algebra. For example, given two matrices A and B with entries in some field $\mathbb{K}$, we may consider them to be equivalent if the rows of A generate the same subspace as the rows of B. Mathematically, the two matrices are equivalent if there exists an invertible matrix U with entries in $\mathbb{K}$ such that $UA = B$. Thus, A can be transformed into B by a sequence of elementary row operations. The normal form commonly used is the row echelon form. If B is in row echelon form, we can easily determine the rank and nullity, a basis for the row space, and bases for the left and right nullspace. We can also easily solve the system of linear equations associated with the matrix A.

In linear algebra, it is also useful to consider two matrices to be equivalent if they represent the same linear transformation up to a change of coordinates. In this case, A and B are equivalent if there exists an invertible change of coordinate matrix P such that $P^{-1}AP = B$. The normal forms in this case include the Jordan canonical form and the Frobenius form (also known as the rational canonical form) [31]. A matrix in the Jordan canonical form is a block diagonal matrix that gives the eigenvalues, and the corresponding transformation matrix gives the generalized eigenvectors. The Frobenius form is also block diagonal, such that its blocks are the companion matrices of its invariant factors. Other invariants such as the minimal polynomial, the characteristic polynomial, and other properties derivable from eigenvalues can easily be obtained once A has been transformed into these forms.

We can also have normal forms when the entries do not come from a field. In computer algebra, we often encounter matrices with entries from a Euclidean domain $\mathbb{D}$, such as the integers $\mathbb{Z}$ or the polynomial ring $\mathbb{K}[x]$ over some field $\mathbb{K}$. We may define two matrices to be equivalent if their rows generate the same $\mathbb{D}$-module (or lattice). One useful normal form is the Hermite normal form, which is triangular and has size restrictions on the off-diagonal entries. Any matrix A can be transformed into a matrix in Hermite normal form H, so that there exists an invertible transformation matrix U such that $UA = H$. In this case, U is invertible in $\mathbb{D}$, so that U^{-1} exists and has entries in $\mathbb{D}$. Such matrices are called unimodular matrices. The Hermite form is useful for solving systems of linear diophantine equations. It can also be used to compute one-sided greatest common divisors of polynomial matrices, a topic discussed in this thesis.

Another notion of equivalence for matrices over a Euclidean domain $\mathbb{D}$ considers two matrices A and B to be equivalent if one can be transformed into another by means of both elementary row and column operations. That is, there exist unimodular matrices U and V satisfying $UAV = B$. Any matrix can be transformed into a diagonal form known as the Smith normal form, which reveals the invariant factors of the matrix [31]. It is also useful for solving systems of linear equations [34, 35].

In this thesis, we consider matrices whose entries are polynomials and, more generally, Ore polynomials [52]. Ore polynomials are generalizations of linear differential operators, linear difference operators, and ordinary polynomials. They differ from ordinary polynomials in that multiplication is not commutative. We consider two matrices to be equivalent if their rows generate the same module. Some useful operations on such matrices are division and the computation of one-sided greatest common divisor and least common multiple. One difficulty is that the set of matrices do not form an integral domain. If we consider these matrices as univariate polynomials with matrix coefficients, we cannot easily perform these operations in the same way as ordinary polynomials because the leading coefficient may be singular.

We study algorithms to transform these matrices into equivalent ones whose "leading coefficients" are nonsingular. These include the row-reduced (or column-reduced if we consider two matrices to be equivalent if their columns generate the same module) form [41], the weak Popov form [50], and the Popov form [41]. Informally, the leading coefficient with respect to row degrees has full row rank for a matrix in row-reduced form, while it is in upper echelon form for a matrix in weak Popov form. A matrix in the Popov form is in weak Popov form, and additionally its leading coefficient with respect to column degrees is the identity matrix. The leading coefficient with respect to row degrees is natural when we consider taking polynomial combinations of the rows of the matrix. In some cases, we can also reverse the coefficients to obtain algorithms to transform matrices so that the trailing coefficient is nonsingular. The applications of these normal forms are considered in Section 1.2.

Example 1.1 *Consider the following matrices with entries in $\mathbb{Z}[z]$.*

$$\mathbf{A}(z) = \begin{bmatrix} z^4 + z^3 + 10z + 4 & 4z^4 + 2z^3 + z^2 \\ z^3 + 10 & 4z^3 + z^2 \end{bmatrix} \quad \mathbf{B}(z) = \begin{bmatrix} z^3 + 4 & z^3 + z^2 \\ z^3 + 10 & 4z^3 + z^2 \end{bmatrix}$$

$$\mathbf{C}(z) = \begin{bmatrix} z^3 + 4 & z^3 + z^2 \\ 6 & 3z^3 \end{bmatrix} \quad \mathbf{D}(z) = \begin{bmatrix} z^3 + 2 & z^2 \\ 2 & z^3 \end{bmatrix}.$$

The leading coefficient with respect to row degrees of $\mathbf{A}(z)$ *is* $\begin{bmatrix} 1 & 4 \\ 1 & 4 \end{bmatrix}$, *which is singular. Subtracting z times the second row from the first row gives* $\mathbf{B}(z)$ *with leading coefficient* $\begin{bmatrix} 1 & 1 \\ 1 & 4 \end{bmatrix}$, *which is nonsingular. Thus,* $\mathbf{B}(z)$ *is in row-reduced* *form. Similarly, subtracting the first row from the second row gives* $\mathbf{C}(z)$ *with leading coefficient* $\begin{bmatrix} 1 & 1 \\ 0 & 3 \end{bmatrix}$*, so that* $\mathbf{C}(z)$ *is in weak Popov form (and hence row-reduced). Finally,* $\mathbf{D}(z)$ *can be obtained from* $\mathbf{C}(z)$ *by row operations. Its leading coefficient with respect to column degrees is the identity matrix. Thus,* $\mathbf{D}(z)$ *is in Popov form.* $\square$

1.1 Computational Challenges

The row-reduced form and Popov form are obtained classically by performing invertible row operations [41]. However, these classical algorithms often perform poorly when coefficient growth is a concern. Although the size of the coefficients in the input and the final output is often small, the size of the coefficients in the intermediate results can grow exponentially. Since the complexity of arithmetic operations (e.g. addition and multiplication of ring elements) depends on the size of the operands, it is not sufficient to analyze only the number of arithmetic operations performed by an algorithm. Instead, we must analyze the number of bit operations performed.

Example 1.2 *Consider the matrix*

$$A = \begin{bmatrix} -20 & 0 & -7 & -196 & 49 \\ 9 & 1 & 2 & -4 & 0 \\ -1 & 0 & 1 & 32 & -7 \\ -46 & -5 & -10 & 21 & 0 \\ 81 & 10 & 20 & -35 & 1 \end{bmatrix} \in \mathbb{Z}^{5\times 5}.$$

It turns out that $\det A = 1$ *and so* A *is unimodular. Therefore, its Hermite form is the identity matrix and its entries are small. If we eliminate the first two columns using the extended Euclidean algorithm, the intermediate result is*

$$\begin{bmatrix} 1 & 0 & -109 & -3376 & 763 \\ 0 & 1 & 11 & 284 & -63 \\ 0 & 0 & -27 & -836 & 189 \\ 0 & 0 & -4969 & -153855 & 34783 \\ 0 & 0 & 8739 & 270581 & -61172 \end{bmatrix}.$$

The size of the coefficients in the intermediate result has increased significantly.

Hafner and McCurley gave a more impressive example where a 20×20 *matrix with entries between 0 and 10 gave a Hermite form with an entry exceeding* 10^{5011} *[39].* □

Example 1.3 *Consider the matrix*

$$\mathbf{A}(z) = \begin{bmatrix} -5z^2+15z+1 & -z^2+z+6 & 0 & 0 & z-3 \\ 8z^2+5z-3 & -9z-8 & 2z+1 & 0 & 3 \\ 8z+4 & -6 & 2 & 0 & 0 \\ -4z-2 & -2z^2-6z-7 & z & 1 & 2z+2 \\ -8z^2-21z-6 & 5z+7 & -2z-3 & 0 & 1 \end{bmatrix} \in \mathbb{Q}[z]^{5\times 5}.$$

Here, $\det \mathbf{A}(z) = 2$ *and again,* $\mathbf{A}(z)$ *is unimodular. Thus, its Hermite normal form is a diagonal matrix whose nonzero entries are either* 1 *or* 2. *After eliminating the first two columns, the intermediate result can be presented as a matrix of pairs* (α, β) *where* α *is the degree and* β *is the number of decimal digits in the largest coefficient of the polynomial entry:*

$$\begin{bmatrix} (0,1) & (-\infty,0) & (5,20) & (-\infty,0) & (4,23) \\ (-\infty,0) & (0,1) & (2,12) & (-\infty,0) & (1,15) \\ (-\infty,0) & (-\infty,0) & (3,2) & (-\infty,0) & (2,3) \\ (-\infty,0) & (-\infty,0) & (6,21) & (0,1) & (5,21) \\ (-\infty,0) & (-\infty,0) & (7,23) & (-\infty,0) & (6,24) \end{bmatrix}.$$

Notice that both the degree and the coefficient size of the entries increase. Again, this growth is significant for larger matrices. □

In this thesis, we study algorithms for computing the row-reduced form, the weak Popov form, and the Popov form for polynomial matrices (also known as matrix polynomials). The row-reduced form and the weak Popov form are also considered for matrices of shift polynomials, which are special cases of Ore polynomials. For the general case of matrices of Ore polynomials our algorithm cannot be used to obtain these normal forms. Nevertheless, our algorithm can be used to perform row operations to determine the rank and a row-reduced basis for the left nullspace. In the case of polynomial matrices our algorithms can also be used to compute the corresponding normal forms based on column operations. The algorithms considered in this thesis compute the normal forms (and the corresponding transformation matrices) via row operations while controlling the growth of the coefficients in the intermediate results. We concentrate our study for the case when the coefficients of the polynomials in the matrix are integers or multivariate polynomials, which covers many applications in computer algebra. In this case, naïve implementations of the classical algorithms can lead to exponential growth in the size of the intermediate results [36]. As a result, these algorithms may fail to compute the answer due to a lack of resources (either time or space). When coefficient growth is not a concern (e.g. when the coefficients are elements of a finite field), other efficient algorithms can also be used [1, 3, 4, 11, 16, 38, 41, 50, 51, 53].

The Fast Fraction-Free Gaussian (FFFG) elimination algorithm by Beckermann and Labahn [12] is the starting point of our study. We formulate the computation of the normal forms in terms of finding certain solutions of linear systems of equations over the coefficient field. While standard techniques for solving linear system of equations can be applied [6, 36], our algorithms take advantage of the special structure in the coefficient matrix to perform Gaussian elimination efficiently. We then apply fraction-free and modular techniques to control the coefficient growth in the intermediate expressions. By studying the associated linear systems of equations, we obtain bounds on the size of intermediate results leading to bounds on complexity. Moreover, our algorithms are general in that no special properties on the input are assumed (e.g. full rank).

1.2 Applications

Row-reduced (and column-reduced) form and Popov form of polynomial matrices have numerous applications in control theory. For example, the differential equations describing multivariable systems can be transformed into algebraic descriptions by the Laplace transform. Hence, the transfer functions can be modelled by matrix-fraction descriptions (MFDs) represented by ratios of polynomial matrices [8, 41, 43].

Example 1.4 *Let $u_1(t)$ and $u_2(t)$ be the inputs of a linear system, and $y_1(t)$ and $y_2(t)$ be the outputs. Suppose that the inputs and outputs are related by the differential equations*

$$\begin{aligned}
y_1''(t) + 5y_1(t) + y_2'(t) - 5y_2(t) &= 2u_1'(t) + u_1(t) + 3u_2''(t) + 3u_2(t)\\
y_1'(t) + 5y_1(t) + 3y_2''(t) + y_2(t) &= 3u_1''(t) + u_1(t) + u_2'(t) + u_2(t).
\end{aligned}$$

Applying the Laplace transform we get the system (assuming that the initial condi-

tions are zero)

$$\begin{bmatrix} s^2+5 & s-5 \\ s+5 & 3s^2+1 \end{bmatrix} \cdot \begin{bmatrix} Y_1(s) \\ Y_2(s) \end{bmatrix} = \begin{bmatrix} 2s+1 & 3s^2+3 \\ 3s^2+1 & s+1 \end{bmatrix} \cdot \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix}.$$

Thus,

$$\begin{bmatrix} Y_1(s) \\ Y_2(s) \end{bmatrix} = \begin{bmatrix} s^2+5 & s-5 \\ s+5 & 3s^2+1 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 2s+1 & 3s^2+3 \\ 3s^2+1 & s+1 \end{bmatrix} \cdot \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix},$$

so that the outputs are represented by a matrix fraction multiplied by the inputs.

□

When the numerator and the denominator are relatively prime, one can obtain a minimal state-space realization of the system. That is, one obtains a model with the fewest number of states that realize the input-output relation. Algorithms to compute a column-reduced form or the Popov form can be used to compute the greatest common (one-sided) divisor (GCD) of the numerator and the denominator and remove the common factor [12, 21, 41]. These algorithms can also be used to compute a one-sided least common multiple (LCM) and a minimal polynomial basis. Furthermore, they can be used to solve related problems such as the minimal partial realization problem [41], which finds the shortest matrix recurrence relation for a sequence of matrices.

Example 1.5 *Let* $\mathbf{A}(z)$ *and* $\mathbf{B}(z)$ *be two matrices having the same number of* *columns. If we form the matrix*

$$\mathbf{F}(z) = \begin{bmatrix} \mathbf{A}(z) \\ \mathbf{B}(z) \end{bmatrix}$$

and compute the row-reduced form, we obtain a matrix of the form

$$\begin{bmatrix} \mathbf{G}(z) \\ \mathbf{0} \end{bmatrix}.$$

It can be shown that $\mathbf{G}(z)$ *is a greatest common right divisor of* $\mathbf{A}(z)$ *and* $\mathbf{B}(z)$ *(see Section 4.5).* □

Different types of state-space realization can be obtained depending on whether the denominator is in row-reduced (and column-reduced) form or in Popov form. In addition, these normal forms have the advantage that the row degrees are minimal among all equivalent matrices. These normal forms also ensure that the leading coefficient with respect to row degrees is nonsingular, which is a standard assumption if one wishes to perform division on polynomial matrices [26, 54, 55, 56]. We also point out that if we "shift" the input matrix and compute the Popov form, we obtain the shifted Popov form which includes the Hermite form [14, 15].

Transforming a polynomial matrix into weak Popov form and Popov form can also be viewed as lattice reduction in the module generated by the rows of the matrix [50]. This has applications in factoring bivariate polynomials.

Matrices of Ore polynomials can be used to represent systems of linear differential equations, difference equations, and other generalizations. By performing row operations on the matrices to obtain a normal form, we can determine the rank and obtain a row-reduced basis of the left nullspace of such matrices [9]. This allows us to determine if an inhomogeneous system of equations has any solution.

Matrices of shift polynomials (a special case of Ore polynomials) can be used to represent systems of linear recurrence equations [1, 2, 3, 4, 7, 9, 10]. By transforming such matrices into row-reduced form, one can obtain bounds on the degrees of the numerator and the denominator of any rational solution of the system. The method of undetermined coefficients can then be used to solve for the solution. This approach can also be used to find rational solutions of linear functional systems, which include linear systems of equations containing linear differential and difference operators as well as other generalizations.

Example 1.6 *As an example, we show how to obtain degree bounds of polynomial solutions of a homogeneous system of linear differential equations with polynomial coefficients. We represent a polynomial as a sequence of its coefficients, and the action of the operators on this sequence can be represented by shift operators on the sequences. For example, multiplication by x corresponds to shifting the coefficient of x^n to the coefficient of x^{n+1}; differentiation corresponds to shifting the coefficient of x^n to the coefficient of x^{n-1} while multiplying by n. A system of differential equations can then be represented as a system of linear recurrences on the coefficient sequences*

$$\mathbf{R}(n, Z) \cdot X_n = \mathbf{0},$$

where $\mathbf{R}(n, Z)$ is a square matrix of linear shift operators, Z is the shift operator, and $X = \{X_i\}_{i=0}^{\infty}$ is a sequence of the coefficients of the polynomials. If we write

$\mathbf{R}(n, Z) = \sum_{i=0}^{N} R_i(n) Z^i$ *where* $R_i(n)$ *are polynomial matrices in* n*, we get*

$$R_0(n) \cdot X_n = -\sum_{i=1}^{N} R_i(n) Z^i \cdot X_n = -\sum_{i=1}^{N} R_i(n) \cdot X_{n+i}, \tag{1.1}$$

By reversing coefficients, our algorithm can be used to transform the system into an equivalent one such that $R_0(n)$ *is nonsingular for all* $n \geq K$ *for some* K *that can be easily obtained from* R_0*. In particular, we can set* K *to be larger than the largest integer root of* $\det R_0$*, which is not identically zero because* R_0 *is nonsingular. This allows us to "solve" the coefficient* X_n *in terms of* X_{n+i}*. If* $X_{n+i} = 0$ *for all* $i > 0$ *and* $R_0(n)$ *is nonsingular, it follows from (1.1) that* $X_n = 0$*. Therefore, if a polynomial solution of degree* D *exists then* $K \geq D$*. Hence,* K *can be used as a degree bound on the polynomial solutions.* □

1.3 Computation Techniques: Fraction-Free and Modular Algorithms

In this section, we examine two methods to control growth in the coefficient size in intermediate results. While the size of the coefficients in the input and the required normal form is often small, the size of coefficients in intermediate results can grow significantly. Any efficient algorithm must control this growth. We will use Gaussian elimination to illustrate these methods.

Row operations are often used to eliminate an entry in the matrix. For example, given the matrix

$$A_0 = \begin{bmatrix} a & b & c & \cdots & \cdots \\ d & e & f & \cdots & \cdots \\ g & h & i & \cdots & \cdots \\ \vdots & \vdots & \vdots & & \end{bmatrix} \in \mathbb{D}^{m \times s},$$

we can subtract d/a times the first row from the second row to eliminate the first column. Notice that fractions are introduced and the simplification of fractions involve hidden computations (e.g. GCD computations). Alternatively, we may multiply the second row by a and then subtract d times the first row. Applying this to all rows gives

$$A_1 = \begin{bmatrix} a & b & c & \cdots & \cdots \\ 0 & ae - bd & af - cd & \cdots & \cdots \\ 0 & ah - bg & ai - cg & \cdots & \cdots \\ \vdots & \vdots & \vdots & & \end{bmatrix}.$$

If we perform Gaussian elimination with these row operations, we have *division-free* Gaussian elimination. The entries of the transformed matrix remain in $\mathbb{D}$, but the size of the entries may double after each step. This leads to an algorithm with an exponential complexity. GCD computations can be performed to remove common factors in each row, but then there is little advantage of using division-free Gaussian elimination.

If we perform one more step of elimination to eliminate the second column, we get

$$A_2 = \begin{bmatrix} a & b & c & \cdots & \cdots \\ 0 & ae-bd & af-cd & \cdots & \cdots \\ 0 & 0 & a(\cdots) & \cdots & \cdots \\ \vdots & \vdots & \vdots & & \end{bmatrix},$$

where every element in the last $m-2$ rows is divisible by a. Therefore, we can remove the common factor without any GCD computation. Continuing this way, it can be shown that the entries in the last $m-k$ rows of A_k are divisible by ${A_{k-1}}^{(k-1,k-1)}$, the pivot element used in the previous step. A known common factor is easily predicted without any GCD computation. This is known as the *fraction-free Gaussian elimination* [6, 36]. It can be shown that this is the largest possible factor removed in general (i.e. if the entries of the matrix are distinct indeterminates), and that all intermediate results obtained during the algorithm can be represented as minors of the input matrix. This gives a polynomial bound on the size of the intermediate results, leading to an efficient algorithm.

Another method to control growth is to use modular homomorphisms to map the problem into other domains in which coefficient growth is not an issue (or less severe) [36]. The results computed under a number of different modular homomorphisms are used to reconstruct the final answer using the Chinese remainder theorem. For example, if $\mathbb{D} = \mathbb{Z}$, we can perform the computations over the finite field $\mathbb{Z}_p$ where p is a prime. If $\mathbb{D} = \mathbb{Z}[x]$ we may apply an evaluation homomorphism and perform the computation in $\mathbb{Z}$. Instead of performing Gaussian elimination on A over $\mathbb{Z}$, we may perform the elimination over $\mathbb{Z}_p$ for several primes p and reconstruct the result by Chinese remaindering. An algorithm using this approach is called a *modular algorithm*.

There are two main issues in modular algorithms. First, the result computed under a modular homomorphism must be the homomorphic image of the desired result. If the answer is not unique then we need to choose a normalization to ensure that the results computed under different modular homomorphisms correspond to the same answer in the original domain. However, it is possible that the result computed under a modular homomorphism is not the image of the desired result, regardless of the normalization chosen. For example, a matrix that is nonsingular over the field of fractions of $\mathbb{D}$ may become singular when a modular homomorphism is applied. We call such a modular homomorphism *unlucky*, and the computed result is typically discarded. Secondly, a bound on the size of the coefficients in the result needs to be established. This allows us to obtain a bound on the number of homomorphic images required to reconstruct the final result. These two issues are usually dealt with using linear algebra.

Fraction-free and modular algorithms have also been used successfully for computing GCDs of polynomials and Ore polynomials [19, 20, 28, 29, 36, 44, 45, 47].

1.4 Overview

The remaining chapters of this thesis are organized as follows.

In Chapter 2, we define the mathematical objects of interest and the notation used in this thesis. We also briefly review some of the existing approaches.

In Chapter 3, we consider the problem of performing row reduction of a matrix of Ore polynomials in a fraction-free way. This allows us to determine the rank and a row-reduced basis for the left nullspace of a matrix of Ore polynomials. This work has been reported in [9].

In Chapter 4, we show that the algorithm in the previous chapter guarantees additional properties when it is applied to matrices of shift polynomials (which include ordinary polynomials). Using these properties we obtain a fraction-free algorithm for computing a weak Popov form. This also leads to a fraction-free algorithm for computing one-sided GCD and LCM that generalizes the classical subresultant theory [19, 20, 28, 44, 45]. This work has been reported in [9, 10].

In Chapter 5, we consider an alternate approach to control coefficient growth. Based on the algorithm given in the previous chapter, we develop a modular algorithm for computing a row-reduced form of a polynomial matrix. We define lucky primes and give a bound on the number of primes needed to reconstruct the final result. We also examine how we can make use of the results computed under an unlucky prime in some cases. Part of this work has been reported in [24].

In Chapter 6, we give a modular algorithm for computing the Popov form of a polynomial matrix. We define lucky primes based on the definition of lucky primes in the previous algorithm. We also give a bound on the size of the coefficients in the final result, leading to a bound on the number of primes required for reconstruction.

Finally, in Chapter 7 we give some concluding remarks and a discussion on future research directions.

Chapter 2

Preliminaries

In this chapter we give definitions of the mathematical objects of interest in this thesis. We also briefly review existing approaches for computing normal forms of matrices of polynomials and Ore polynomials.

2.1 Basic Definitions

Linear differential equations are often studied in terms of the associated differential operator [18]. For example, the differential equation

$$y''(x) - (2x + 3)y'(x) + (6x - 2)y(x) = 0$$

can be rewritten in terms of a linear differential operator as

$$(D^2 - (2x + 3)D + (6x - 2))y(x) = 0,$$

where D denotes differentiation with respect to the independent variable x. We can view linear differential operators as polynomials in D, with the exception that multiplication by D obeys the product rule

$$(Df)g = D(fg) = fg' + f'g = (fD + f')g.$$

Therefore,

$$Df = fD + f'. \tag{2.1}$$

Algebraic operations (e.g. factoring) on linear differential operators can help in determining the solutions of the equation. A similar multiplication rule is used when dealing with operators arising from linear recurrence equations or other similar equations. For example, the recurrence equation

$$y_{n+2} - (2n+3)y_{n+1} + (6n-2)y_n = 0$$

can be represented by the operator

$$E^2 - (2n+3)E + (6n-2)$$

where E denotes the shift operator with respect to n. In this case, we have

$$(Ef)g = E(fg) = (Ef)(Eg).$$

Therefore,

$$Ef = (Ef)E. \tag{2.2}$$

Ore polynomials [52] allow us to study all such operators in a unified way.

Definition 2.1 (Ore ring) *Let $\mathbb{D}$ be an integral domain and $\mathbb{Q}_{\mathbb{D}}$ be its field of fractions. The set of polynomials $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$ in Z is an* Ore ring*, with σ (the* conjugate*) an automorphism on $\mathbb{Q}_{\mathbb{D}}$, and δ (the σ-*derivation*) an additive homomorphism on $\mathbb{Q}_{\mathbb{D}}$ such that the ring multiplication obeys the rule*

$$Z \cdot a = \sigma(a) \cdot Z + \delta(a)$$

for all $a \in \mathbb{Q}_{\mathbb{D}}$. □

Notice that by expanding both sides of $Z \cdot (ab) = (Z \cdot a) \cdot b$, we see that δ satisfies

$$\delta(ab) = \sigma(a)\delta(b) + \delta(a)b.$$

Example 2.2 *Let $\mathbb{K}$ be a field. Some examples of Ore rings are*

(a) $\mathbb{D} = \mathbb{K}[x]$ with Z the differential operator and $\sigma(f(x)) = f(x)$, $\delta(f(x)) = \frac{d}{dx}f(x)$. Therefore, $Z \cdot f(x) = f(x) \cdot Z + f'(x)$. This models linear differential operators as shown in (2.1);

(b) $\mathbb{D} = \mathbb{K}[n]$ with Z the shift operator such that $\sigma(f(n)) = f(n+1)$ and $\delta = 0$. This models linear shift operators as shown in (2.2);

(c) $\mathbb{D} = \mathbb{K}[x]$ *with* Z *the q-differentiation operator. In this case,* $\sigma(f(x)) = f(qx)$ *and* $\delta(f(x)) = \frac{f(qx)-f(x)}{qx-x}$*;*

(d) $\mathbb{D} = \mathbb{K}[n, q]$ *with* Z *the q-shift operator. In this case,* $\sigma(f(n)) = f(qn)$ *and* $\delta(f(n)) = 0$*.*

(e) $\mathbb{D} = \mathbb{K}[x]$ *with* Z *the Eulerian operator. Here,* $\sigma(f(x)) = f(x)$ *and* $\delta(f(x)) = xf'(x)$*;*

(f) $\mathbb{D} = \mathbb{K}[x]$ *where* Z *is the Mahlerian operator and let* $p > 1$ *be an integer. Then,* $\sigma(f(x)) = f(x^p)$ *and* $\delta = 0$*.*

More examples can be found, for example, in [25]. □

When $\delta = 0$, we have the *ring of shift polynomials*[1] and we use the notation $\mathbb{Q}_{\mathbb{D}}[Z; \sigma]$. If $\sigma = 1_{\mathbb{Q}_{\mathbb{D}}}$, the identity function on $\mathbb{Q}_{\mathbb{D}}$, and $\delta = 0$, then we have the usual commutative polynomial ring $\mathbb{Q}_{\mathbb{D}}[Z]$. We will often use z instead of Z to emphasize that the indeterminate commutes with the coefficients under multiplication, and denote the *ring of polynomials* as $\mathbb{Q}_{\mathbb{D}}[z]$. We note that by a suitable change of variables, we may always transform an Ore ring into an equivalent in which σ is the identity function or δ is the zero function [27], but the transformation may introduce fractions even if the original Ore polynomials have coefficients in $\mathbb{D}$.

[1]We note that some authors call these "skew polynomials" (or "skew Laurent polynomial" if negative powers are allowed), while other authors use the term "skew polynomials" synonymously with "Ore polynomials." We will use the term "shift polynomials" to avoid confusion.

Given an $m \times s$ matrix of Ore polynomials $\mathbf{F}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times s}$, we denote by $\mathcal{M}_{\mathbf{F}(Z)}$ the $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-module generated by the rows of $\mathbf{F}(Z)$. That is,

$$\begin{aligned}\mathcal{M}_{\mathbf{F}(Z)} &= \left\{ q_1(Z)\cdot\mathbf{F}(Z)^{(1,\cdot)} + \cdots + q_m(Z)\cdot\mathbf{F}(Z)^{(m,\cdot)} : q_i(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta] \right\} \\ &= \left\{ \mathbf{Q}(Z)\cdot\mathbf{F}(Z) : \mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m} \right\},\end{aligned}$$

where $\mathbf{F}(Z)^{(i,\cdot)}$ denotes the ith row of $\mathbf{F}(Z)$. The *left nullspace* of $\mathbf{F}(Z)$, denoted $\mathcal{N}_{\mathbf{F}(Z)}$, is the $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-module defined as

$$\mathcal{N}_{\mathbf{F}(Z)} = \left\{ \mathbf{V}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m} : \mathbf{V}(Z)\cdot\mathbf{F}(Z) = \mathbf{0} \right\}.$$

We also define the *rank* of the matrix $\mathbf{F}(Z)$, denoted rank $\mathbf{F}(Z)$, to be the maximum number of $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-linearly independent rows of $\mathbf{F}(Z)$. We remark that our definition of rank is different from (and perhaps simpler than) that of [3, 4] or [27] who consider the rank of the $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-module generated by the rows of $\mathbf{F}(Z)$ or the rank of the matrix over the skew field of left fractions of elements in $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$. These quantities are shown to be equivalent in [9].

We are interested in performing two types of elementary row operations on $\mathbf{F}(Z)$. An *elementary row operation of the first type*, or simply *elementary row operation*, is one of the following operations:

(a) interchange two rows;

(b) multiply a row by a nonzero element in $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$;

(c) add a polynomial multiple of one row to another.

An *elementary row operation of the second type* is one of the following:

(a) interchange two rows;

(b) multiply a row by a nonzero element in $\mathbb{Q}_{\mathbb{D}}$;

(c) add a polynomial multiple of one row to another.

The difference between the two types of operations is the multiplier allowed in (b). We note that elementary row operations of the first type are not necessarily invertible over $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$. Row operations are also called *row reductions*.

Formally, we can view a sequence of elementary row operations of the first type as a matrix $\mathbf{U}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times m}$ with the result of these row operations given by $\mathbf{T}(Z) = \mathbf{U}(Z)\cdot\mathbf{F}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times s}$. For row operations of the second type $\mathbf{U}(Z)$ has the additional property that there exists a left inverse $\mathbf{V}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times m}$ such that $\mathbf{V}(Z)\cdot\mathbf{U}(Z) = \mathbf{I}_m$. It can be shown that $\mathbf{V}(Z)$ is also a right inverse of $\mathbf{U}(Z)$ [9]. We say that $\mathbf{U}(Z)$ is *unimodular* if $\mathbf{U}(Z)$ has an inverse.

We can similarly define elementary column operations. Sequences of column operations correspond to multiplication by a transformation matrix on the right. In this thesis we will study column operations only for polynomial matrices[2], and describe our algorithms in terms of row operations. In this case, we may perform column operations by performing row operations on the transpose of $\mathbf{F}(z)$ and then taking the transpose of the results. The normal forms studied in this thesis satisfy

[2]For matrices of Ore polynomials, we can use a similar technique by applying row operations to the adjoint $\mathbf{F}(Z)^*$, where $(\mathbf{F}(Z)^*)^{(i,j)} = (\mathbf{F}(Z)^{(j,i)})^*$ and * is the adjoint of an Ore polynomial [5]. However, the adjoint of an Ore polynomial is only defined in some cases, and the usefulness of the corresponding column normal forms is unclear. Therefore we will only consider row operations and normal forms based on row operations for matrices of Ore polynomials in this thesis.

the property that a matrix is in normal form defined in terms of row operations if and only if its transpose is in the corresponding normal form defined in terms of column operations. One can, of course, reformulate the algorithms in terms of column operations for efficiency.

2.2 Notation

We shall adopt the following conventions for this thesis. We denote the ring of integers $\mathbb{Z}$ and the field of rational numbers $\mathbb{Q}$. For any prime $p \in \mathbb{Z}$, we denote by $\mathbb{Z}_p$ the finite field of p elements.

We assume that $\mathbf{F}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z; \sigma, \delta]^{m \times s}$. Let $N = \deg \mathbf{F}(Z)$, and write

$$\mathbf{F}(Z) = \sum_{j=0}^{N} F_j Z^j, \text{ with } F_j \in \mathbb{Q}_{\mathbb{D}}^{m \times s}.$$

We also write $c_j(\mathbf{F}(Z)) = F_j$ as the coefficient of Z^j in $\mathbf{F}(Z)$. We denote the elements of $\mathbf{F}(Z)$ by $\mathbf{F}(Z)^{(k,\ell)}$, and the elements of F_j by $F_j{}^{(k,\ell)}$. The ith row of $\mathbf{F}(Z)$ is denoted $\mathbf{F}(Z)^{(i,\cdot)}$ and the jth column is denoted $\mathbf{F}(Z)^{(\cdot,j)}$. For any sets of row and column indices I and J, $\mathbf{F}(Z)^{(I,\cdot)}$ is the submatrix of $\mathbf{F}(Z)$ consisting of the rows indexed by I, $\mathbf{F}(Z)^{(\cdot,J)}$ is the submatrix of $\mathbf{F}(Z)$ consisting of the columns indexed by J, and $\mathbf{F}(Z)^{(I,J)}$ is the submatrix of $\mathbf{F}(Z)$ consisting of the rows and columns indexed by I and J.

For any vector of integers (also called multi-index) $\vec{\omega} = (\vec{\omega}^{(1)}, \ldots, \vec{\omega}^{(p)})$, we denote by $|\vec{\omega}| = \sum_{i=1}^{p} \vec{\omega}^{(i)}$. When $<$, $\leq$, $>$, and $\geq$ are used to compare vectors it is understood that the relationship is true if and only if it is true for each pair of components in the vectors. Similarly max and min give the vectors whose components are the maximum and minimum of the corresponding components of their input vectors. Additionally, two vectors can be compared in lexicographical order. We say that $\vec{v} \leq_{lex} \vec{w}$ if $\vec{v} = \vec{w}$ or if the leftmost nonzero entry in $\vec{v} - \vec{w}$ is negative. The vector $\vec{e}_i$ denotes the ith unit vector (of the appropriate dimension) such that $\vec{e}_i^{(i)} = 1$ and $\vec{e}_i^{(j)} = 0$ for $j \neq i$; we also have $\vec{e} = (1, \ldots, 1)$ (of the appropriate dimension). We denote by $\mathbf{I}_m$ the $m \times m$ identity matrix, and by $Z^{\vec{\omega}}$ the matrix of Ore polynomials having $Z^{\vec{\omega}^{(i)}}$ on the diagonal and 0 everywhere else.

A matrix of Ore polynomials $\mathbf{F}(Z)$ is said to have *row degree* $\vec{\nu} = \text{rdeg } \mathbf{F}(Z)$ (*column degree* $\vec{\mu} = \text{cdeg } \mathbf{F}(Z)$) if the ith row has degree $\vec{\nu}^{(i)}$ (the jth column has degree $\vec{\mu}^{(j)}$). The *leading coefficient* of $\mathbf{F}(Z)$, denoted $\text{LC}\,(\mathbf{F}(Z))$, is F_N. The *leading row coefficient*, denoted $\text{LC}_{row}\,(\mathbf{F}(Z))$, is defined as $\text{LC}\left(Z^{N\cdot\vec{e} - \text{rdeg } \mathbf{F}(Z)} \cdot \mathbf{F}(Z)\right)$, and the *leading column coefficient*, denoted $\text{LC}_{col}\,(\mathbf{F}(Z))$, is defined as $\text{LC}_{row}\left(\mathbf{F}(Z)^T\right)^T$ if $\mathbf{F}(Z)$ is a polynomial matrix.

Example 2.3 *Let*

$$\mathbf{A}(Z) = \begin{bmatrix} nZ^2 + 2 & (n-1)Z^2 \\ nZ & (n-1)Z - 3 \end{bmatrix}.$$

Here $N = 2$ and rdeg $\mathbf{A}(Z) = (2, 1)$. If $\mathbf{A}(Z) \in \mathbb{Q}(n)[Z;\sigma]^{2\times 2}$ such that $\sigma(a(n)) = a(n+1)$, then

$$\begin{aligned} &LC_{row}\,(\mathbf{A}(Z)) = LC\left(Z^{(0,1)} \cdot \mathbf{A}(Z)\right) \\ =&LC\left(\begin{bmatrix} nZ^2 + 2 & (n-1)Z^2 \\ (n+1)Z^2 & nZ^2 - 3Z \end{bmatrix}\right) = \begin{bmatrix} n & n-1 \\ n+1 & n \end{bmatrix}. \end{aligned}$$

On the other hand, if we consider $\mathbf{A}(Z) \in \mathbb{Q}(n)[z]^{2\times 2}$*, then*

$$LC_{row}\left(\mathbf{A}(Z)\right) = \begin{bmatrix} n & n-1 \\ n & n-1 \end{bmatrix}.$$

$\square$

Remark 2.4 *The leading row coefficient is defined in this manner because we are interested in the elements of* $\mathcal{M}_{\mathbf{F}(Z)}$*. That is, we wish to examine elements of the form* $\mathbf{Q}(Z) \cdot \mathbf{F}(Z)$ *for some* $\mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$*. If* $\vec{\mu} = rdeg\,\mathbf{F}(Z)$ *and* $d = \max_{1\le j\le m}\left\{\deg \mathbf{Q}(Z)^{(1,j)} + \vec{\mu}^{(j)}\right\}$*, then it is useful if we can guarantee that* $\deg(\mathbf{Q}(Z)\cdot\mathbf{F}(Z)) = d$*. In other words, we need to guarantee that the coefficient of* Z^d *in* $\mathbf{Q}(Z)\cdot\mathbf{F}(Z)$ *does not vanish. This coefficient can be written in terms of the leading row coefficient as*

$$\begin{aligned}
&\sum_{j=1}^{m} {Q_{d-\vec{\mu}^{(j)}}}^{(1,j)} \cdot Z^{d-\vec{\mu}^{(j)}} \cdot \mathbf{F}(Z)^{(j,\cdot)} \\
=&\sum_{j=1}^{m} {Q_{d-\vec{\mu}^{(j)}}}^{(1,j)} \cdot Z^{d-N} \cdot Z^{N-\vec{\mu}^{(j)}} \cdot \mathbf{F}(Z)^{(j,\cdot)} \\
=&\begin{bmatrix} {Q_{d-\vec{\mu}^{(j)}}}^{(1,1)} & \cdots & {Q_{d-\vec{\mu}^{(j)}}}^{(1,m)} \end{bmatrix} \cdot Z^{d-N} \cdot LC_{row}\left(\mathbf{F}(Z)\right).
\end{aligned}$$

This allows us to easily predict the degree of $\mathbf{Q}(Z)\cdot\mathbf{F}(Z)$ *from the degrees of* $\mathbf{Q}(Z)$ *and* $\mathbf{F}(Z)$ *in a similar way as in the case of scalar polynomials, provided that* $LC_{row}\left(\mathbf{F}(Z)\right)$ *satisfies additional properties (see Lemma 3.3).* $\square$

2.3 Normal Forms

A normal form is simply a representative chosen from a class of equivalent objects. An object in normal form usually has some desirable properties. For example, it may be easy to obtain from the normal form invariants for all equivalent objects. In this thesis, we focus on row-equivalent matrices of Ore polynomials. We consider two matrices of Ore polynomials to be equivalent if their rows generate the same $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-module. In other words, two matrices $\mathbf{A}(Z)$ and $\mathbf{B}(Z)$ are equivalent if there exists a unimodular matrix $\mathbf{U}(Z)$ such that $\mathbf{A}(Z) = \mathbf{U}(Z) \cdot \mathbf{B}(Z)$.

We first give the definition of the row-reduced (column-reduced) form.

Definition 2.5 *A matrix* $\mathbf{F}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times s}$ *is in* row-reduced form *(or* $\mathbf{F}(z)$ *is* row-reduced*) if rank* $LC_{row}(\mathbf{F}(Z)) = m$*. If* $\mathbf{F}(z) \in \mathbb{Q}_{\mathbb{D}}[z]^{m\times s}$ *then* $\mathbf{F}(z)$ *is in* column-reduced form *(or* $\mathbf{F}(z)$ *is column-reduced) if rank* $LC_{col}(\mathbf{F}(z)) = s$*. Here, the rank of a matrix is defined over* $\mathbb{Q}_{\mathbb{D}}$*.* □

Properties of polynomial matrices in row-reduced and column-reduced forms are well known [41]. Some of these properties for row-reduced forms are extended to matrices of Ore polynomials in [9].

Example 2.6 *Let* $\mathbf{A}(Z)$ *be the matrix of Ore polynomials defined in Example 2.3. If* $\mathbf{A}(Z) \in \mathbb{Q}(n)[Z;\sigma]^{2\times 2}$ *such that* $\sigma(a(n)) = a(n+1)$*, then* $LC_{row}(\mathbf{A}(Z))$ *is nonsingular, so that* $\mathbf{A}(Z)$ *is row-reduced. On the other hand, if we consider* $\mathbf{A}(Z) \in \mathbb{Q}(n)[z]^{2\times 2}$*, then* $LC_{row}(\mathbf{A}(Z))$ *is singular and so* $\mathbf{A}(Z)$ *is not row-reduced.*

□

Before we define the Popov form, we first define a normal form called the weak

Popov form (also called quasi-Popov form [14]). The weak Popov form is often the intermediate form obtained when one wishes to compute the Popov form from a matrix in row-reduced form [50].

Definition 2.7 *A matrix* $\mathbf{F}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times s}$ *is in* weak Popov form *if the leading row coefficient of the submatrix formed from the nonzero rows of* $\mathbf{F}(Z)$ *is in upper echelon form (up to row permutation). In other words, if we define the pivot index of row* i*, denoted* Π_i*, to be*

$$\Pi_i = \begin{cases} \min_{1\le j\le s}\left\{j : \deg \mathbf{F}(Z)^{(i,j)} = \deg \mathbf{F}(Z)^{(i,\cdot)}\right\} & \mathbf{F}(Z)^{(i,\cdot)} \neq \mathbf{0} \\ 0 & \mathbf{F}(Z)^{(i,\cdot)} = \mathbf{0} \end{cases},$$

then $\Pi_i \neq \Pi_j$ *whenever* $i \neq j$*, and* $\mathbf{F}(Z)^{(i,\cdot)}$ *and* $\mathbf{F}(Z)^{(j,\cdot)}$ *are both nonzero.* □

A matrix in weak Popov form is also row-reduced if there are no zero rows. We can also define the weak Popov form in terms of the leading column coefficient for polynomial matrices. We are now ready to define the Popov form.

Definition 2.8 *A matrix* $\mathbf{F}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times s}$ *is in* Popov form *if it is in weak Popov form, and for all* i *such that* $\mathbf{F}(Z)^{(i,\cdot)}$ *is nonzero,*

(a) $\mathbf{F}(Z)^{(i,\Pi_i)}$ *is monic;*

(b) $\deg \mathbf{F}(Z)^{(j,\Pi_i)} < \deg \mathbf{F}(Z)^{(i,\Pi_i)}$ *for all* $j \neq i$*.* □

A Popov form based on column operations can also be defined for polynomial matrices.

Example 2.9 *The matrix* $\mathbf{A}(Z)$ *in Example 2.3 is not in weak Popov form because* $\Pi_1 = \Pi_2 = 1$. *Let*

$$\mathbf{P}(Z) = \begin{bmatrix} nZ^2 + 2 & (n-1)Z^2 \\ Z & (n-1)Z^3 - 3 \end{bmatrix}.$$

Here, $\Pi_1 = 1$ *and* $\Pi_2 = 2$ *and so* $\mathbf{P}(Z)$ *is in weak Popov form. Furthermore, the degree constraints in Definition 2.8(b) are satisfied. Thus, the matrix*

$$\mathbf{P}^*(Z) = \begin{bmatrix} \frac{1}{n} & 0 \\ 0 & \frac{1}{n-1} \end{bmatrix} \cdot \mathbf{P}(Z)$$

is in Popov form. □

Remark 2.10 *We note that whether the matrix* $\mathbf{P}(Z)$ *in Example 2.9 is in weak Popov form does not depend on whether the entries are considered to be Ore polynomials, shift polynomials, or ordinary polynomials. This is true in general because* σ *is an automorphism on the coefficient field, so that the upper echelon structure of the leading coefficient is unaffected by the application of* σ. *Similarly, whether* $\mathbf{P}^*(Z)$ *is in Popov form does not depend on* σ. □

Any matrix $\mathbf{F}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z; \sigma, \delta]^{m \times s}$ can be transformed into one of the above normal forms by means of elementary row operations of the second type. Neither the row-reduced form nor the corresponding transformation matrix is unique. However, it can be shown that the row degree of the row-reduced form is minimal, and is unique up to permutation. The weak Popov form and the corresponding transformation matrix are also not unique, but the set of pivot indices are unique. Finally, the Popov form is unique (up to row permutation) for any input matrix

$\mathbf{F}(Z)$, but the transformation matrix is only unique when $\mathbf{F}(Z)$ has full row rank. Otherwise, the transformation matrix is unique only if we impose additional degree constraints on its elements [15].

Remark 2.11 *The Popov form provides a useful tool since it is unique among all matrices equivalent under elementary row operations. It has the further advantage that the row degree is minimized. On the other hand, although the Hermite form is also unique, the degrees of its entries can be large. The Popov form is most useful if we are only interested in having a "nice" leading coefficient. It is useful for determining if rows of two matrices generate the same module. It is not useful for solving systems of linear diophantine equations because a matrix in Popov form is not triangular.* □

2.4 Special Matrices Related to Row Operations

In this section we define the striped Krylov matrix, which is a tool that allows us to study row operations on matrices of Ore polynomials in terms of linear algebra. By reformulating row operations as linear systems of equations over $\mathbb{Q}_{\mathbb{D}}$, it is possible to apply standard tools from linear algebra such as determinants and fraction-free Gaussian elimination [6] to study row operations.

We represent row operations on the matrix $\mathbf{F}(Z)$ as multiplication by a matrix of Ore polynomials $\mathbf{U}(Z)$ on the left. Writing the result of the row operations as $\mathbf{T}(Z) = \mathbf{U}(Z) \cdot \mathbf{F}(Z)$, we see that each row of $\mathbf{T}(Z)$ can be written as a polynomial combination of the rows of $\mathbf{F}(Z)$. If $\vec{\mu} = \text{cdeg}\, \mathbf{U}(Z)$, we can write the ith row of

$\mathbf{T}(Z)$ as

$$\mathbf{T}(Z)^{(i,\cdot)} = \sum_{j=1}^{m} \mathbf{U}(Z)^{(i,j)} \cdot \mathbf{F}(Z)^{(j,\cdot)} = \sum_{j=1}^{m} \sum_{k=0}^{\vec{\mu}^{(j)}} {U_k}^{(i,j)} \cdot Z^k \cdot \mathbf{F}(Z)^{(j,\cdot)}. \tag{2.3}$$

To study this equation using linear algebra, we rewrite (2.3) into an equation over $\mathbb{Q}_{\mathbb{D}}$. We define for any $\mathbf{A}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times s}$ and multi-index $\vec{\omega}$ the corresponding matrix

$$\mathbf{A}_{\vec{\omega}} = \left[{A_0}^{(\cdot,1)} \quad \cdots \quad {A_{\vec{\omega}^{(1)}}}^{(\cdot,1)} \quad | \quad \cdots \quad | \quad {A_0}^{(\cdot,m)} \quad \cdots \quad {A_{\vec{\omega}^{(m)}}}^{(\cdot,m)} \right].$$

We also define

$$\mathbf{K}(\vec{\mu}, \mathbf{F}(Z)) = \begin{bmatrix} \mathbf{F}(Z)^{(1,\cdot)} \\ \vdots \\ Z^{\vec{\mu}^{(1)}} \cdot \mathbf{F}(Z)^{(1,\cdot)} \\ \vdots \\ \mathbf{F}(Z)^{(m,\cdot)} \\ \vdots \\ Z^{\vec{\mu}^{(m)}} \cdot \mathbf{F}(Z)^{(m,\cdot)} \end{bmatrix}.$$

This allows us to write (2.3) as

$$\mathbf{T}(Z)^{(i,\cdot)} = {\mathbf{U}_{\vec{\mu}}}^{(i,\cdot)} \cdot \mathbf{K}(\vec{\mu}, \mathbf{F}(Z)), \tag{2.4}$$

where ${\mathbf{U}_{\vec{\mu}}}^{(i,\cdot)}$ is a vector over $\mathbb{Q}_{\mathbb{D}}$. We see from (2.4) that

$$ {\mathbf{T}_{\vec{\omega}}}^{(i,\cdot)} = {\mathbf{U}_{\vec{\mu}}}^{(i,\cdot)} \cdot \mathbf{K}(\vec{\mu}, \mathbf{F}(Z))_{\vec{\omega}}. \tag{2.5}$$

Row operations are often done to eliminate certain coefficients. Thus, if we wish to eliminate the first $\vec{\omega}^{(j)}$ coefficients of $\mathbf{T}(Z)^{(i,j)}$ it is equivalent to solving the linear system of equations (over $\mathbb{Q}_{\mathbb{D}}$)

$$\mathbf{U}_{\vec{\mu}}{}^{(i,\cdot)} \cdot K(\vec{\mu}, \vec{\omega}, \mathbf{F}(Z)) = \mathbf{0} \tag{2.6}$$

for some multi-index $\vec{\mu}$, where $K(\vec{\mu}, \vec{\omega}, \mathbf{F}(Z)) := \mathbf{K}(\vec{\mu}, \mathbf{F}(Z))_{\vec{\omega}-\vec{e}}$, such that if $\vec{\omega}^{(j)} = 0$, column j is not present in $K(\vec{\mu}, \vec{\omega}, \mathbf{F}(Z))$.

Definition 2.12 *The matrix $K(\vec{\mu}, \vec{\omega}, \mathbf{F}(Z))$ is called the* striped Krylov matrix of degree $\vec{\mu}$ and order $\vec{\omega}$ for $\mathbf{F}(Z)$. *When $\mathbf{F}(Z)$ is clear from the context, we will simply write $K(\vec{\mu}, \vec{\omega})$.* $\square$

Notice the striped Krylov matrix can be thought of as having m stripes, each corresponding to a row of $\mathbf{F}(Z)$. The first row in each stripe gives the coefficients of $\mathbf{F}(Z)$ in the corresponding row, and each successive row is obtained by multiplying the previous row by Z on the left while ignoring the higher order terms introduced. The structure inherent in the striped Krylov matrix will be exploited in our algorithms.

Example 2.13 *Let $a(z)$, $b(z) \in \mathbb{Q}_{\mathbb{D}}[z]$ of degrees n_1 and n_2, respectively, such that $n_1 \geq n_2$. Consider the reciprocal polynomials $a^*(z) = z^{n_1} \cdot a(1/z)$ and $b^*(z) = z^{n_2} \cdot b(1/z)$. If $\mathbf{F}(z) = [a^*(z), b^*(z)]^T$ is a 2×1 polynomial matrix, we see that $K((n_2 - 1, n_1 - 1), n_1 + n_2)$ is the well-known Sylvester matrix [36].* $\square$

Example 2.14 *Let $\vec{\mu} = \vec{\omega} = (2, 2)$, and*

$$\mathbf{F}(Z) = \begin{bmatrix} 2Z^2 + 2xZ + x^3 & Z^2 - Z + (2x+1) \\ (x-1)Z + 2 & 3xZ - x \end{bmatrix} \in \mathbb{Q}(x)[Z; \sigma, \delta]^{2\times 2},$$

with $\sigma(a(x)) = a(x)$ and $\delta(a(x)) = \frac{d}{dx}a(x)$. Then

$$K(\vec{\mu}, \vec{\omega}, \mathbf{F}(Z)) = \left[\begin{array}{ccc|ccc} x^3 & 2x & 2 & 2x+1 & -1 & 1 \\ 3x^2 & x^3+2 & 2x & 2 & 2x+1 & -1 \\ 6x & 6x^2 & x^3+4 & 0 & 4 & 2x+1 \\ \hline 2 & x-1 & 0 & -x & 3x & 0 \\ 0 & 3 & x-1 & -1 & -x+3 & 3x \\ 0 & 0 & 4 & 0 & -2 & -x+6 \end{array}\right].$$

□

Example 2.15 *We can write $K(\vec{\mu}, \vec{\omega})$ as a matrix consisting of $m \times s$ blocks B_{ij}, such that $K(\vec{\mu}, \vec{\omega}) = [B_{ij}]$ where B_{ij} is a $(\vec{\mu}^{(i)}+1)\times\vec{\omega}^{(j)}$ block. If $\mathbf{F}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z; \sigma]^{m\times s}$ and $\vec{\omega}^{(j)} = k$, then*

$$B_{ij} = \begin{bmatrix} \sigma^0({F_0}^{(i,j)}) & \sigma^0({F_1}^{(i,j)}) & \sigma^0({F_2}^{(i,j)}) & \cdots & \cdots & \sigma^0({F_{k-1}}^{(i,j)}) \\ 0 & \sigma^1({F_0}^{(i,j)}) & \sigma^1({F_1}^{(i,j)}) & \cdots & \cdots & \sigma^1({F_{k-2}}^{(i,j)}) \\ \vdots & \ddots & \ddots & \ddots & & \vdots \\ 0 & \cdots & 0 & \sigma^{\vec{\mu}^{(i)}}({F_0}^{(i,j)}) & \cdots & \sigma^{\vec{\mu}^{(i)}}({F_{k-\vec{\mu}^{(i)}-1}}^{(i,j)}) \end{bmatrix}.$$

□

The striped Krylov matrix is rectangular in general, and we also need to define particular submatrices that are often useful in analyzing pivoting schemes.

Definition 2.16 *Let J be the lexicographically smallest set of column indices such that $K(\vec{\nu}, \vec{\omega}, \mathbf{F}(Z))^{(\cdot,J)}$ has full column rank for some $\vec{\nu}$. The matrix $K^*(\vec{\mu}, \vec{\omega}, \mathbf{F}(Z))$ is defined to be the submatrix $K(\vec{\mu}, \vec{\omega}, \mathbf{F}(Z))^{(\cdot,J)}$.* $\square$

Intuitively, $K^*(\vec{\mu}, \vec{\omega})$ removes from $K(\vec{\mu}, \vec{\omega})$ the columns that are zero (and hence do not require elimination) if one performs Gaussian elimination to eliminate the columns in order, regardless of the pivoting scheme chosen. We note that $K^*(\vec{\mu}, \vec{\omega})$ is square if rank $K(\vec{\mu}, \vec{\omega}) = |\vec{\mu} + \vec{e}|$.

2.5 Previous Approaches

In this section we give a brief overview of other approaches for computing the normal forms we are interested in.

2.5.1 Direct Methods

In the first group of algorithms [1, 3, 4, 11, 41, 50], elementary row operations of the second type are used to eliminate unwanted coefficients until the desired properties are satisfied. The row operations are chosen such that the process eventually terminates. For example, a polynomial matrix $\mathbf{F}(z)$ can be transformed into row-reduced form by repeatedly finding a nonzero vector in the left kernel of the leading row coefficient of the intermediate result [11]. That is, a vector $\vec{w} \neq \vec{0}$ such that

$$\vec{w} \cdot \mathrm{LC}_{row}(\mathbf{F}(z)) = \vec{0},$$

where we assume that $\vec{w}^{(k)} = 0$ if $\mathbf{F}(z)^{(k,\cdot)}$ is the zero row. Such a vector exists if and only if $\mathbf{F}(z)$ is not in row-reduced form. Let $\vec{\mu} = \operatorname{rdeg} \mathbf{F}(z)$ and choose a row k such that $\vec{\mu}^{(k)}$ is maximal among the rows j with $\vec{w}^{(j)} \neq 0$. We can apply the following elementary row operations of the second type represented by the identity matrix with the kth row replaced,

$$\mathbf{Q}(z) = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ \vec{w}^{(1)} \cdot z^{\vec{\mu}^{(k)} - \vec{\mu}^{(1)}} & \cdots & \vec{w}^{(k)} & \cdots & \vec{w}^{(m)} \cdot z^{\vec{\mu}^{(k)} - \vec{\mu}^{(m)}} \\ & & & \ddots & \\ & & & & 1 \end{bmatrix}, \tag{2.7}$$

which reduces the degree of the kth row. Therefore, this process must eventually terminate and we obtain a polynomial matrix in row-reduced form. The same approach can be extended to matrices of Ore polynomials [9] (see Theorem 3.1). As suggested in [3, 4], the vector $\vec{w}$ in the above algorithm could be chosen in $\mathbb{D}^{1\times m}$ by performing fraction-free Gaussian elimination on $\mathrm{LC}_{row}\left(\mathbf{F}(z)\right)$ [6], leading to a fraction-free algorithm for row-reducing a matrix of Ore polynomials. However, extraneous factors are not removed between two steps. In order to prevent an exponential growth of coefficients, it would still be necessary to remove the content of the rows of the intermediate results during the computations, an operation which could be very expensive.

Similarly, a weak Popov form and the Popov form can be obtained by eliminating high order coefficients by elementary row operations of the second type. These algorithms are often sufficient when coefficient growth is not a concern (e.g. if the coefficients of the polynomials come from a finite field). However, there is no explicit control of coefficient growth in other cases.

2.5.2 Indirect Methods

Other algorithms obtain the normal forms indirectly by solving systems of linear equations instead of applying elementary row operations. They typically make use of algorithms for computing a solution with special properties, such as a minimal polynomial basis, and extract the normal form from the solution. It is well known that the rows of a polynomial matrix form a minimal polynomial basis for the module generated by the rows if and only if the polynomial matrix is row-reduced [41]. For example, to find $\mathbf{U}(z) \cdot \mathbf{F}(z) = \mathbf{T}(z)$ such that $\mathbf{T}(z)$ is in row-reduced form, a minimal polynomial basis of the left nullspace of the matrix $[\mathbf{F}(z)^T \cdot z^b,\ -\mathbf{I}_m]^T$ is computed for a sufficiently large b [16, 38, 51]. The minimal polynomial basis can then be written as $[\mathbf{U}(z),\ \mathbf{T}(z) \cdot z^b]$ where $\mathbf{T}(z) \cdot z^b$ (and hence $\mathbf{T}(z)$) is row-reduced because $\mathbf{U}(z)$ does not contribute to the leading row coefficient of $[\mathbf{U}(z),\ \mathbf{T}(z) \cdot z^b]$ when b is sufficiently large. Such a “shift” in the input is also used in [14, 15] to compute the Popov form of a polynomial matrix indirectly. The indirect methods are often used in a numerical setting (i.e. with floating-point numbers) because the algorithms used for solving the system of equations have desirable numerical properties. A disadvantage of these methods is that even if the input matrix is already in normal form, the algorithm cannot detect this easily and must perform all of its calculations. In addition, both the degree and the dimensions of the input are increased because of the shift and the augmented matrix.

2.5.3 Fraction-free Method

The Fast Fraction-free Gaussian (FFFG) elimination algorithm of Beckermann and Labahn [12] is an algorithm that can be used to perform row operations on polynomial matrices in a fraction-free way. The algorithm has been extended to compute the row-reduced form for polynomial matrices [13]. We give a very brief overview of the FFFG elimination algorithm here. The details of this algorithm can also be found as a special case of the algorithm in Chapter 3 and Chapter 4, where we generalize the algorithm for matrices of Ore polynomials and shift polynomials.

Roughly speaking, the FFFG elimination algorithm performs fraction-free elimination of a polynomial matrix by performing fraction-free Gaussian elimination [6] on the corresponding striped Krylov matrix. However, it exploits the structure inherent in the striped Krylov matrix to make the elimination more efficient. The FFFG algorithm uses the polynomial representation of the rows and performs elimination only on the rows corresponding to $z^{\vec{\mu}^{(i)}} \cdot \mathbf{F}(z)^{(i,\cdot)}$ from the ith stripe for each i, where $\vec{\mu}$ is a multi-index recording how many times some row in each stripe has been used as a pivot. If a row in the ith stripe is chosen as the pivot row, then the row corresponding to the new $\vec{\mu}$ in the ith stripe has to be computed. This is done by multiplying the pivot row by z followed by "degree adjustments". This ensures that we obtain the same result as if fraction-free Gaussian elimination is applied to the striped Krylov matrix. As in the fraction-free Gaussian elimination algorithm, the pivot element used in the previous elimination step is a common factor of the intermediate results and can be removed. Furthermore, the intermediate results can be represented as minors of the striped Krylov matrix so that bounds on the size of the coefficients can be obtained using Hadamard's inequality [40].

We remark that Bitmead *et al.* proposed a numerical algorithm (i.e. for floating-point numbers) that performs Gaussian elimination on a "generalized Sylvester matrix," which is similar to the striped Krylov matrix we define [17, 41]. In this case, the intermediate results in the next stripe are computed simply by multiplying by z without the degree adjustments. Gentle [37] proposed a fraction-free version of this algorithm provided that the desired rows can always be chosen as pivot. A fraction-free algorithm for the general case is not known.

Chapter 3

Fraction-free Row Reduction of Matrices of Ore Polynomials

In this chapter we give a fraction-free algorithm to perform row reductions on matrices of Ore polynomials. Using this algorithm we show how to compute the rank and a row-reduced basis of the left nullspace of a matrix of Ore polynomials.

3.1 Some Results on Matrices of Ore Polynomials

We first give some results on matrices of Ore polynomials, which generalize the well-known results for polynomial matrices [41]. These results are necessary in the development of our algorithm and are not found elsewhere (except in [9]). First, we prove that any matrix of Ore polynomials can be transformed into row-reduced form. We also give degree bounds on the entries of the transformation matrix. The degree bounds will be used to determine the number of steps required in our algorithm.

Theorem 3.1 *For any* $\mathbf{F}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times s}$ *there exists a unimodular matrix* $\mathbf{U}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times m}$*, with* $\mathbf{T}(Z) = \mathbf{U}(Z)\cdot\mathbf{F}(Z)$ *having* $r \le \min\{m,s\}$ *nonzero rows,* $\textit{rdeg}\,\mathbf{T}(Z) \le \textit{rdeg}\,\mathbf{F}(Z)$*, and where the submatrix consisting of the r nonzero rows of* $\mathbf{T}(Z)$ *is row-reduced. Moreover, the unimodular multiplier satisfies the degree bound*

$$\textit{rdeg}\,\mathbf{U}(Z) \le \vec{\nu} + (|\vec{\mu}| - |\vec{\nu}| - \alpha)\cdot\vec{e} \tag{3.1}$$

where $\vec{\mu} := \max(\vec{0}, \textit{rdeg}\,\mathbf{F}(Z))$*,* $\vec{\nu} := \max(\vec{0}, \textit{rdeg}\,\mathbf{T}(Z))$*, and* $\alpha = \min_j\{\vec{\mu}^{(j)}\}$*.*

Proof. We construct $\mathbf{U}(Z)$ and $\mathbf{T}(Z)$ in a way similar to that for polynomial matrices as described in Section 2.5.1. In addition, we will also verify the degree bound (3.1) at each step of the construction.

Starting with $\mathbf{U}(Z) = \mathbf{I}_m$ and $\mathbf{T}(Z) = \mathbf{F}(Z)$, we construct a sequence of unimodular matrices $\mathbf{U}(Z)$ and $\mathbf{T}(Z) = \mathbf{U}(Z)\cdot\mathbf{F}(Z)$. The degree bound (3.1) is clearly satisfied initially.

To compute the results in the next step, $\mathbf{U}(Z)_{new}$ and $\mathbf{T}(Z)_{new}$, denote by J the set of indices of zero rows of $\mathbf{T}(Z)$, and $L = \mathrm{LC}_{row}(\mathbf{T}(Z))$. If the matrix formed by the nonzero rows of $\mathbf{T}(Z)$ is not row-reduced, we can find $\vec{w} \in \mathbb{Q}_{\mathbb{D}}^{1\times m}$ with $\vec{w} \ne \vec{0}$, $\vec{w}\cdot L = \vec{0}$, and $\vec{w}^{(j)} = 0$ for $j \in J$. We choose the index of the updated row k as before. Following (2.7) we define $\mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$ to be the matrix

$$\begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ \sigma^{\vec{\nu}^{(k)}-t}\left(\vec{w}^{(1)}\right)\cdot Z^{\vec{\nu}^{(k)}-\vec{\nu}^{(1)}} & \cdots & \sigma^{\vec{\nu}^{(k)}-t}\left(\vec{w}^{(i)}\right) & \cdots & \sigma^{\vec{\nu}^{(k)}-t}\left(\vec{w}^{(m)}\right)\cdot Z^{\vec{\nu}^{(k)}-\vec{\nu}^{(m)}} \\ & & & \ddots & \\ & & & & 1 \end{bmatrix}.$$

If we define $\mathbf{U}(Z)_{new} = \mathbf{Q}(Z)\cdot\mathbf{U}(Z)$ and $\mathbf{T}(Z)_{new} = \mathbf{Q}(Z)\cdot\mathbf{T}(Z)$, then

$$\begin{aligned}
\mathbf{T}(Z)_{new}^{(k,\cdot)} &= \mathbf{Q}(Z)^{(k,\cdot)}\cdot\mathbf{T}(Z) \\
&= \sum_{\vec{w}^{(j)}\neq 0} \sigma^{\vec{\nu}^{(k)}-t}(\vec{w}^{(j)})\cdot Z^{\vec{\nu}^{(k)}-\vec{\nu}^{(j)}}\cdot T_{\vec{\nu}^{(j)}}{}^{(j,\cdot)}\cdot Z^{\vec{\nu}^{(j)}} + \text{ lower degree terms} \\
&= \sum_{j=1}^{m} \sigma^{\vec{\nu}^{(k)}-t}(\vec{w}^{(j)})\sigma^{\vec{\nu}^{(k)}-\vec{\nu}^{(j)}}(T_{\vec{\nu}^{(j)}}{}^{(j,\cdot)})\cdot Z^{\vec{\nu}^{(k)}} + \text{ lower degree terms} \\
&= \sigma^{\vec{\nu}^{(k)}-t}(\vec{w}\cdot L)\cdot Z^{\vec{\nu}^{(k)}} + \text{ lower degree terms,}
\end{aligned}$$

where $t = \deg \mathbf{T}(Z)$. Hence $\deg \mathbf{T}(Z)_{new}^{(k,\cdot)} \leq \vec{\nu}^{(k)} - 1$, showing that $\text{rdeg } \mathbf{T}(Z)_{new} \leq \text{rdeg } \mathbf{T}(Z)$. Since $\mathbf{Q}(Z)^{(k,k)} \neq 0$ by construction, we may consider $\mathbf{W}(Z)$ obtained from $\mathbf{I}_m$ by replacing its (k,j) entry by $-\left(\mathbf{Q}(Z)^{(k,k)}\right)^{-1}\cdot\mathbf{Q}(Z)^{(k,j)}$ for $j \neq k$, and by $\left(\mathbf{Q}(Z)^{(k,k)}\right)^{-1}$ for $j = k$. It can easily be verified that $\mathbf{W}(Z)\cdot\mathbf{Q}(Z) = \mathbf{Q}(Z)\cdot\mathbf{W}(Z) = \mathbf{I}_m$. Thus, $\mathbf{U}(Z)_{new}$ is also unimodular. Making use of the degree bounds for $\mathbf{U}(Z)$, we also get that $\deg(\mathbf{Q}(Z)^{(k,\cdot)}\cdot\mathbf{U}(Z)) \leq \vec{\nu}^{(k)} + |\vec{\mu}| - |\vec{\nu}| - \alpha$.

Hence the degree bounds for $\mathbf{U}(Z)_{new}$ are obtained by observing that

$$\text{rdeg}\,\mathbf{U}(Z)_{new} \leq \vec{\nu} + (|\vec{\mu}| - |\vec{\nu}| - \alpha) \cdot \vec{e} \leq \vec{\nu}_{new} + (|\vec{\mu}| - |\vec{\nu}_{new}| - \alpha) \cdot \vec{e}.$$

Finally, we notice that, in each step of the algorithm, we either produce a new zero row in $\mathbf{T}(Z)$, or else decrease $|\vec{\nu}|$, the sum of the row degrees of nonzero rows of $\mathbf{T}(Z)$, by at least one. Hence the procedure terminates, which implies that the nonzero rows of $\mathbf{T}(Z)$ form a row-reduced submatrix. $\square$

Remark 3.2 *There is an example [15, Example 5.6] of a polynomial matrix* $\mathbf{F}(z)$ *which is unimodular (and hence* $\mathbf{T}(Z) = \mathbf{I}$*), has row degree* $N \cdot \vec{e}$*, and where its multiplier satisfies rdeg* $\mathbf{U}(Z) = (m-1)N \cdot \vec{e}$*. Hence the worst case estimate of Theorem 3.1 for the degree of* $\mathbf{U}(Z)$ *is sharp.* $\square$

In fact, the quantity r of Theorem 3.1 equals rank $\mathbf{F}(Z)$. Before we prove this result, we need some essential properties of row-reduced matrices that are well known for polynomial matrices (e.g. see [41]).

Lemma 3.3

(a) Let $\mathbf{F}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times s}$*, with* $\vec{\mu} = $ *rdeg* $\mathbf{F}(Z)$*.* $\mathbf{F}(Z)$ *is row-reduced if and only if, for any* $\mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$*,*

$$\deg \mathbf{Q}(Z) \cdot \mathbf{F}(Z) = \max_{1\leq j\leq m} \left\{ \vec{\mu}^{(j)} + \deg \mathbf{Q}(Z)^{(1,j)} \right\}.$$

(b) Let $\mathbf{A}(Z) = \mathbf{B}(Z)\cdot\mathbf{C}(Z)$ *be matrices of Ore polynomials of sizes* $m\times s$*,* $m\times r$*, and* $r \times s$*, respectively. Then rank* $\mathbf{A}(Z) \leq r$*.*

(c) Let $\mathbf{A}(Z) = \mathbf{B}(Z) \cdot \mathbf{C}(Z)$ be as in part (b), with $\mathbf{A}(Z)$ and $\mathbf{C}(Z)$ row-reduced and row degrees $\vec{\alpha}^{(1)} \leq \vec{\alpha}^{(2)} \leq \ldots \leq \vec{\alpha}^{(m)}$ and $\vec{\gamma}^{(1)} \leq \vec{\gamma}^{(2)} \leq \ldots \leq \vec{\gamma}^{(r)}$, respectively. Then $m \leq r$, and $\vec{\alpha}^{(j)} \geq \vec{\gamma}^{(j)}$ for $j = 1, \ldots, m$.

(d) Let $\mathbf{T}(Z) = \mathbf{U}(Z) \cdot \mathbf{S}(Z)$, such that $\mathbf{U}(Z)$ is unimodular and both $\mathbf{S}(Z)$ and $\mathbf{T}(Z)$ are row-reduced. Then, up to permutation, the row degrees of $\mathbf{S}(Z)$ and $\mathbf{T}(Z)$ coincide.

Proof. For any $\mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$, let $N' := \max_{1\leq j\leq m}\left\{\vec{\mu}^{(j)} + \deg \mathbf{Q}(Z)^{(1,j)}\right\}$. Let $\vec{h} \in \mathbb{Q}_{\mathbb{D}}^{1\times m}$ be the vector such that

$$\mathbf{Q}(Z)^{(1,j)} = \vec{h}^{(j)} Z^{N'-\vec{\mu}^{(j)}} + \text{ lower degree terms.}$$

Note that $\vec{h} \neq \vec{0}$. Clearly, $\deg \mathbf{Q}(Z) \cdot \mathbf{F}(Z) \leq N'$, with the coefficient at $Z^{N'}$ being given by

$$\sum_{j=1}^{m} \vec{h}^{(j)} \sigma^{N'-\vec{\mu}^{(j)}} (F_{\vec{\mu}^{(j)}}{}^{(j,\cdot)}) = \vec{h} \cdot \sigma^{N'-N}(\mathrm{LC}_{row}(\mathbf{F}(Z))).$$

Since σ is an automorphism on $\mathbb{Q}_{\mathbb{D}}$, the matrix $\mathbf{F}(Z)$ is row-reduced if and only if $\sigma^j(\mathrm{LC}_{row}(\mathbf{F}(Z)))$ is of full row rank for any integer j; that is, if and only if $\vec{h} \cdot \sigma^j(\mathrm{LC}_{row}(\mathbf{F}(Z))) \neq \vec{0}$ for all $\vec{h} \neq \vec{0}$ and all integers j. This in turn holds if and only if $\deg \mathbf{Q}(Z) \cdot \mathbf{F}(Z) = N'$ for any $\mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$. Therefore, (a) holds.

In order to show (b), we may suppose by eliminating a suitable number of rows of $\mathbf{A}(Z)$ and $\mathbf{B}(Z)$ that rank $\mathbf{A}(Z) = m$. Then $\mathcal{M}_{\mathbf{B}(Z)} \subseteq \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times r}$, the latter being a $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-module of rank r. Hence $r \geq \operatorname{rank} \mathcal{M}_{\mathbf{B}(Z)} \geq \operatorname{rank} \mathbf{B}(Z)$. If $m > r$, then $\mathbf{B}(Z)$ has more rows than columns. Thus, by definition of rank $\mathbf{B}(Z)$ there exists a nontrivial $\mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$ with $\mathbf{Q}(Z) \cdot \mathbf{B}(Z) = \mathbf{0}$. Hence,

$\mathbf{Q}(Z) \cdot \mathbf{A}(Z) = \mathbf{0}$, a contradiction to the fact that $\mathbf{A}(Z)$ has full row rank m. Therefore $r \geq m$, as claimed in part (b).

For a proof of part (c), recall first that the rows of the row-reduced $\mathbf{A}(Z)$ are $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-linearly independent by part (a), and hence $m = \text{rank}\,\mathbf{A}(Z) \leq r$ by part (b). Suppose that $\vec{\alpha}^{(j)} \geq \vec{\gamma}^{(j)}$ for $j < k$, but $\vec{\alpha}^{(k)} < \vec{\gamma}^{(k)}$. Part (a) implies that $\deg \mathbf{B}(Z)^{(j,\ell)} \leq \vec{\alpha}^{(j)} - \vec{\gamma}^{(\ell)}$. Since $\vec{\alpha}^{(j)} < \vec{\gamma}^{(k)} \leq \vec{\gamma}^{(\ell)}$ for $j \leq k \leq \ell$, we may conclude that $\mathbf{B}(Z)^{(j,\ell)} = 0$ for $j \leq k \leq \ell$. Thus, the first k rows of $\mathbf{A}(Z)$ are $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-linear combinations of the first $k-1$ rows of $\mathbf{C}(Z)$. From part (b) it follows that the first k rows of $\mathbf{A}(Z)$ are $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-linearly dependent, a contradiction. Hence the assertion of part (c) holds.

Finally, part (d) is obtained by applying part (c) twice, using the fact that $\mathbf{U}(Z)$ is invertible. □

Remark 3.4 *Lemma 3.3(a) is usually known as the* predictable degree property *in the case of polynomial matrices [41].* □

We now prove a theorem on recovering a row-reduced basis of the left nullspace $\mathcal{N}_{\mathbf{F}(Z)}$. This is a crucial result needed in the development of our algorithm in this chapter, as it allows us to obtain a termination criteria and prove the correctness of our algorithm.

Theorem 3.5 *Let $\mathbf{F}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times s}$, $\mathbf{U}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times m}$ be unimodular, and $\mathbf{T}(Z) = \mathbf{U}(Z) \cdot \mathbf{F}(Z)$ having r nonzero rows, such that the nonzero rows of $\mathbf{T}(Z)$ form a row-reduced matrix. Then*

$$r = rank\, \mathcal{M}_{\mathbf{F}(Z)} = rank\, \mathbf{F}(Z) = m - rank\, \mathcal{N}_{\mathbf{F}(Z)}, \tag{3.2}$$

with a basis of $\mathcal{N}_{\mathbf{F}(Z)}$ given by those rows of $\mathbf{U}(Z)$ corresponding to the zero rows of $\mathbf{T}(Z)$.

Moreover, there exists a row-reduced $\mathbf{W}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{(m-r)\times m}$ with rows forming a basis of $\mathcal{N}_{\mathbf{F}(Z)}$, and rdeg $\mathbf{W}(Z) \leq (m-1)N\cdot\vec{e}$.

Proof. We first prove (3.2) with $\mathbf{F}(Z)$ replaced by $\mathbf{T}(Z)$ and then prove (3.2). Denote by J the set of indices of zero rows of $\mathbf{T}(Z)$. For any $\mathbf{P}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$ we have

$$\mathbf{P}(Z)\cdot\mathbf{T}(Z) = \sum_{j\notin J} \mathbf{P}(Z)^{(1,j)}\cdot\mathbf{T}(Z)^{(j,\cdot)}.$$

By Lemma 3.3(a) the rows $\mathbf{T}(Z)^{(j,\cdot)}$ for $j\notin J$ are $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-linearly independent. Therefore $\mathbf{P}(Z)\in\mathcal{N}_{\mathbf{T}(Z)}$ if and only if $\mathbf{P}(Z)^{(1,j)}=0$ for all $j\notin J$. Hence,

$$r = \text{rank }\mathbf{T}(Z) = m - \text{rank }\mathcal{N}_{\mathbf{T}(Z)}.$$

It is easily seen that $r = \text{rank }\mathbf{T}(Z) \leq \text{rank }\mathcal{M}_{\mathbf{T}(Z)} =: \rho$. Now, given ρ elements of $\mathcal{M}_{\mathbf{T}(Z)}$ which are $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-linearly independent, they can be written as rows of the matrix $\mathbf{B}(Z)\cdot\mathbf{T}(Z)$ for some $\mathbf{B}(Z)\in\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{\rho\times m}$. Then rank $\mathbf{B}(Z)\cdot\mathbf{T}(Z) = \rho$ by construction of $\mathbf{B}(Z)$. Since $\mathbf{T}(Z)$ contains only r nonzero rows, we have $\rho = \text{rank }\mathbf{B}(Z)\cdot\mathbf{T}(Z)\leq r$ by Lemma 3.3(b). Thus, $r=\rho$. Consequently, (3.2) holds if $\mathbf{F}(Z)$ is replaced by $\mathbf{T}(Z)$.

Since $\mathbf{U}(Z)$ is unimodular, it has an inverse $\mathbf{V}(Z)\in\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times m}$. Consequently, $\mathbf{Q}(Z)\in\mathcal{N}_{\mathbf{F}(Z)}$ if and only if $\mathbf{P}(Z)=\mathbf{Q}(Z)\cdot\mathbf{V}(Z)\in\mathcal{N}_{\mathbf{T}(Z)}$. That is,

$$\mathcal{N}_{\mathbf{F}(Z)} = \{\mathbf{P}(Z)\cdot\mathbf{U}(Z) : \mathbf{P}(Z)^{(1,j)} = 0 \text{ for } j\notin J\} = \mathcal{M}_{\mathbf{U}(Z)^{(J,\cdot)}}.$$

Since $\mathbf{U}(Z)$ has a right inverse, we may conclude that $\mathcal{N}_{\mathbf{U}(Z)} = \{\vec{0}\}$, showing that rows of unimodular matrices are linearly independent over $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$. Thus the rows of $\mathbf{U}(Z)^{(J,\cdot)}$ form a basis of $\mathcal{N}_{\mathbf{F}(Z)}$, and

$$m - \text{rank } \mathcal{M}_{\mathbf{F}(Z)} = m - \text{rank } \mathcal{M}_{\mathbf{T}(Z)} = m - r = \text{rank } \mathcal{N}_{\mathbf{F}(Z)}. \tag{3.3}$$

Since again the relation $\rho := \text{rank } \mathbf{F}(Z) \leq \text{rank } \mathcal{M}_{\mathbf{F}(Z)}$ is trivial, for a proof of (3.2) it only remains to show that $\rho < r$ leads to a contradiction. Suppose without loss of generality that the first ρ rows of $\mathbf{F}(Z)$ are linearly independent. Then for any $j = \rho + 1, \ldots, m$, there exists $\mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$ such that

$$\mathbf{Q}(Z)^{(j,j)} \neq 0, \qquad \mathbf{Q}(Z)^{(j,j)} \cdot \mathbf{F}(Z)^{(j,\cdot)} + \sum_{k=1}^{\rho} \mathbf{Q}(Z)^{(j,k)} \cdot \mathbf{F}(Z)^{(k,\cdot)} = 0.$$

This gives $m - \rho$ $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-linearly independent elements of $\mathcal{N}_{\mathbf{F}(Z)}$, contradicting (3.3) that rank $\mathcal{N}_{\mathbf{F}(Z)} = m - r < m - \rho$.

In order to show the second part, suppose that $\mathbf{U}(Z)$ and $\mathbf{T}(Z)$ are those defined in Theorem 3.1. Applying Theorem 3.1 again to $\mathbf{U}(Z)^{(J,\cdot)}$ transforms it into a row-reduced matrix $\mathbf{W}(Z) = \mathbf{V}(Z) \cdot \mathbf{U}(Z)^{(J,\cdot)}$ for some unimodular $\mathbf{V}(Z)$. From the degree bound in Theorem 3.1, we have

$$\deg \mathbf{U}(Z)^{(j,\cdot)} \leq \vec{\nu}^{(j)} - \alpha + (|\vec{\mu}| - |\vec{\nu}|) \leq |\vec{\mu}| - \alpha \leq (m-1)N$$

for all $j \in J$. This gives the desired degree bound on $\mathbf{W}(Z)$. □

Remark 3.6 *The quantity rdeg* $\mathbf{W}(Z)$ *of Theorem 3.5 is an invariant of* $\mathbf{F}(Z)$

since any row-reduced basis of $\mathcal{N}_{\mathbf{F}(Z)}$ has the same row degree (up to permutation) by Lemma 3.3(d). For polynomial matrices, the components of rdeg $\mathbf{W}(Z)$ are referred to as left minimal indices *or* left Kronecker indices *[41, §6.5.4].* □

Finally, we will require a basic property of rank for matrices of Ore polynomials.

Lemma 3.7 *For any $\mathbf{F}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times s}$, rank $\mathbf{F}(Z)$ does not change after applying elementary row operations of the first or second type, or by multiplying $\mathbf{F}(Z)$ on the right by a full rank square matrix of Ore polynomials.*

Proof. Suppose that $\mathbf{A}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{s\times s}$ is of rank s. Then $\mathcal{N}_{\mathbf{A}(Z)} = \{\vec{0}\}$ by (3.2), implying that $\mathcal{N}_{\mathbf{F}(Z)\cdot\mathbf{A}(Z)} = \mathcal{N}_{\mathbf{F}(Z)}$. Hence $\mathbf{F}(Z)\cdot\mathbf{A}(Z)$ and $\mathbf{F}(Z)$ have the same rank by (3.2). If $\mathbf{U}(Z)$ is unimodular, then $\mathcal{M}_{\mathbf{U}(Z)\cdot\mathbf{F}(Z)} = \mathcal{M}_{\mathbf{F}(Z)}$, showing that the rank remains the same after applying elementary row operations of the second type. Finally we need to examine the row operation of multiplying one row of $\mathbf{F}(Z)$ with a nonzero element of $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$. Since $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$ is an integral domain, it is easy to check that $\mathbf{F}(Z)$ and the new matrix will have the same number of $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-linearly independent rows, and hence the same rank. This shows that the rank remains the same after applying elementary row operations of the first type. □

Remark 3.8 *We remark that while the rank remains unchanged under the operations specified in Lemma 3.7, the module generated by the rows of the matrix may be different.* □

3.2 Order Basis

In this section we introduce the notion of order and order bases for a given matrix of Ore polynomials $\mathbf{F}(Z)$. These are the primary tools which will be used for our algorithm.

Informally, we are interested in taking linear combinations of rows of $\mathbf{F}(Z)$ in order to eliminate low order terms, where the number of terms eliminated in each column may be different. Formally such an elimination is captured using the concept of *order*. The components of the order vector gives the number of terms eliminated in each column.

Definition 3.9 *Let $\mathbf{P}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$ be a vector of Ore polynomials and $\vec{\omega}$ a multi-index. Then $\mathbf{P}(Z)$ is said to have* order $\vec{\omega}$ *(with respect to $\mathbf{F}(Z)$) if*

$$\mathbf{P}(Z) \cdot \mathbf{F}(Z) = \mathbf{R}(Z) \cdot Z^{\vec{\omega}} \tag{3.4}$$

with $\mathbf{R}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times s}$. The matrix $\mathbf{R}(Z)$ in (3.4) is called a residual. $\square$

We are interested in *all* possible row operations which eliminate lower order terms of $\mathbf{F}(Z)$. Using our formalism, this corresponds to finding all $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-linear combinations of elements of a given order. This in turn is captured in the definition of an order basis, which gives a basis of the module of all vectors of Ore polynomials having a particular order.

Definition 3.10 *Let $\mathbf{F}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times s}$, and $\vec{\omega}$ and $\vec{\mu}$ be multi-indices. A matrix of Ore polynomials $\mathbf{M}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{m\times m}$ is said to be an* order basis *of order $\vec{\omega}$ and (column) degree $\vec{\mu}$ if*

(a) *every row of* $\mathbf{M}(Z)$ *has order* $\vec{\omega}$,

(b) *every* $\mathbf{P}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$ *of order* $\vec{\omega}$ *can be written as* $\mathbf{P}(Z) = \mathbf{Q}(Z)\cdot\mathbf{M}(Z)$ *for some* $\mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$, *and*

(c) *there exists a nonzero* $d \in \mathbb{Q}_{\mathbb{D}}$ *such that*

$$\mathbf{M}(Z) = d \cdot Z^{\vec{\mu}} + \mathbf{L}(Z)$$

where $\deg \mathbf{L}(Z)^{(k,\ell)} \leq \vec{\mu}^{(\ell)} - 1$.

If in addition $\mathbf{M}(Z)$ *is row-reduced with rdeg* $\mathbf{M}(Z) = \vec{\mu}$, *we say that* $\mathbf{M}(Z)$ *is a* reduced order basis. □

Part (a) of Definition 3.10 states that every row of an order basis eliminates rows of $\mathbf{F}(Z)$ up to a certain order while part (b) implies that the rows describe all eliminates of the order. The intuition of part (c) is that $\vec{\mu}^{(i)}$ gives the number of times row i has been used as a pivot row in a row elimination process. A reduced order basis has added degree constraints, which can be thought of as choosing a specific pivoting strategy.

Remark 3.11 *By the predictable degree property for matrices of Ore polynomials in Lemma 3.3(a), we can show that an order basis is a reduced order basis if and only if rdeg* $\mathbf{M}(Z) = \vec{\mu}$ *and we have the added degree constraint in Definition 3.10(b) that, for all* $j = 1, \ldots, m$,

$$\deg \mathbf{Q}(Z)^{(1,j)} \leq \deg \mathbf{P}(Z) - \vec{\mu}^{(j)}. \tag{3.5}$$

We remark that the definition of order basis given in [10] is slightly more restrictive than our definition of reduced order basis given here. We use the more general definition in order to gain more flexibility with our pivoting. $\square$

Remark 3.12 *In fact, a reduced order basis is a scalar multiple of a matrix in Popov form. Without loss of generality, we may assume that* $0 \leq \vec{\mu}^{(1)} \leq \cdots \leq \vec{\mu}^{(m)}$*. By Definition 3.10(c) it follows that* $\deg \mathbf{M}(Z)^{(k,\Pi_1)} < \vec{\mu}^{(1)}$ *for* $k > 1$*, so that* $\Pi_k \neq \Pi_1$ *for all* $k > 1$*. Applying a similar argument to the remaining rows shows that the pivot indices are unique.* $\square$

A key theorem for proving the correctness of the fraction-free algorithm deals with the uniqueness of order bases. Recall that $\vec{e} = (1, \ldots, 1)$ and $\vec{e}_k$ is the kth unit vector.

Theorem 3.13 *(a) There exists only the trivial row vector* $\mathbf{P}(Z) = 0$ *with column degree* $\leq \vec{\mu} - \vec{e}$ *and order* $\geq \vec{\omega}$*.*

(b) For any $k = 1, \ldots, m$*, there exists a unique row vector with column degree* $\leq \vec{\mu} - \vec{e} + \vec{e}_k$ *and order* $\geq \vec{\omega}$*, up to multiplication with an element from* $\mathbb{Q}_{\mathbb{D}}$*.*

(c) An order basis of a particular order and degree is unique up to multiplication by constants from $\mathbb{Q}_{\mathbb{D}}$*.*

Proof. We only need to show part (a) as (b) and (c) follow directly from (a). Suppose that $\mathbf{P}(Z) \neq 0$ has order $\vec{\omega}$ and column degree $\vec{\mu} - \vec{e}$. By Definition 3.10(b), there exists $\mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$ such that $\mathbf{P}(Z) = \mathbf{Q}(Z) \cdot \mathbf{M}(Z)$. Let j be an index such that $\deg \mathbf{Q}(Z)^{(1,j)}$ is maximum. Since $\mathbf{P}(Z) \neq 0$, it follows that

$\deg \mathbf{Q}(Z)^{(1,j)} \geq 0$. Now,

$$\deg \mathbf{P}(Z)^{(1,j)} = \deg \left(\sum_{k=1}^{m} \mathbf{Q}(Z)^{(1,k)} \cdot \mathbf{M}(Z)^{(k,j)} \right).$$

Note that if $k \neq j$, then

$$\begin{aligned} \deg \left(\mathbf{Q}(Z)^{(1,k)} \cdot \mathbf{M}(Z)^{(k,j)} \right) &= \deg \mathbf{Q}(Z)^{(1,k)} + \deg \mathbf{M}(Z)^{(k,j)} \\ &\leq \deg \mathbf{Q}(Z)^{(1,j)} + \deg \mathbf{M}(Z)^{(k,j)} \\ &\leq \deg \mathbf{Q}(Z)^{(1,j)} + \vec{\mu}^{(j)} - 1. \end{aligned}$$

Also,

$$\deg \mathbf{Q}(Z)^{(1,j)} \cdot \mathbf{M}(Z)^{(j,j)} = \deg \mathbf{Q}(Z)^{(1,j)} + \vec{\mu}^{(j)},$$

so that

$$\deg \mathbf{P}(Z)^{(1,j)} = \deg \mathbf{Q}(Z)^{(1,j)} + \vec{\mu}^{(j)} \geq \vec{\mu}^{(j)}.$$

This contradicts the assumption that $\deg \mathbf{P}(Z)^{(1,j)} \leq \vec{\mu}^{(j)} - 1$. $\square$

We illustrate the notion of order basis with an example related to the pseudo-division of a shift polynomial by another (see [44]). This is well known in the case of ordinary polynomials (see, for example, [49]).

Example 3.14 *Let $a(Z), b(Z) \in \mathbb{D}[Z;\sigma]$ with degrees d_a, d_b, respectively, such that $d_a \geq d_b$. Set $t = d_a - d_b$. We make the substitution $\hat{Z} = Z^{-1}$, $\hat{\sigma} = \sigma^{-1}$, and define the shift polynomials*

$$\overline{a}(\hat{Z}) = a(\hat{Z}^{-1}) \cdot \hat{Z}^{d_a}, \quad \overline{b}(\hat{Z}) = \sigma^{t} \left(b(\hat{Z})^{-1} \cdot \hat{Z}^{d_b} \right), \tag{3.6}$$

where $\sigma\left(\sum_{i=0}^{d_b} b_i \hat{Z}^i\right) := \sum_{i=0}^{d_b} \sigma(b_i)\hat{Z}^i$. *Let* $q(Z), r(Z)$ *be such that*

$$\overline{b}_0^{[t+1]} \cdot a(Z) = q(Z) \cdot b(Z) + r(Z), \tag{3.7}$$

with $\deg q(Z) = t$, $\deg r(Z) < d_b$, *and* $\overline{b}_0^{[t]} := \prod_{i=0}^{t} \hat{\sigma}^i(\overline{b}_0) = \prod_{i=0}^{t} \sigma^i\left(b_{d_b}\right)$. *We define*

$$\overline{q}(\hat{Z}) = q(\hat{Z}^{-1}) \cdot \hat{Z}^t, \quad \overline{r}(\hat{Z}) = r(\hat{Z}^{-1}) \cdot \hat{Z}^{d_b - 1}.$$

Then we can easily verify that

$$\overline{a}(\hat{Z}) = \overline{q}(\hat{Z}) \cdot \overline{b}(\hat{Z}) + \overline{r}(\hat{Z}) \cdot Z^{t+1}.$$

Setting $\overline{\mathbf{F}}(\hat{Z}) = [\overline{a}(\hat{Z}),\ \overline{b}(\hat{Z})]^T$, *we see that*

$$\mathbf{M}(\hat{Z}) = \begin{bmatrix} d & -\overline{q}(\hat{Z}) \\ 0 & d \cdot \hat{Z}^{t+1} \end{bmatrix}, \tag{3.8}$$

satisfies Definition 3.10(a) and (c) with $d = \overline{b}_0^{[t+1]}$, *degree* $\vec{\mu} = (0, t+1)$, *and order* $\vec{\omega} = (t+1)$ *because*

$$\mathbf{M}(\hat{Z}) \cdot \overline{\mathbf{F}}(\hat{Z}) = \begin{bmatrix} d & -\overline{q}(\hat{Z}) \\ 0 & d \cdot \hat{Z}^{t+1} \end{bmatrix} \cdot \begin{bmatrix} \overline{a}(\hat{Z}) \\ \overline{b}(\hat{Z}) \end{bmatrix} = d \cdot \begin{bmatrix} \overline{r}(\hat{Z}) \\ \hat{\sigma}^{t+1}(\overline{b}(\hat{Z})) \end{bmatrix} \cdot \hat{Z}^{t+1}.$$

We will show later (Example 3.19) that Definition 3.10(b) is also satisfied, so that $\mathbf{M}(\hat{Z})$ *is an order basis of degree* $\vec{\mu}$ *and order* $\vec{\omega}$. □

3.3 Determinantal Representations

We are interested in constructing an algorithm for computing recursively order bases $\mathbf{M}(Z)$ for increasing orders. In order to predict the size of these objects and predict common factors, we derive in this section a determinantal representation together with a particular choice of the constant d arising in Definition 3.10(c).

As we have noted in Section 2.4, the elimination of a certain number of low order terms is equivalent to solving a system of linear equations whose coefficient matrix is a striped Krylov matrix. Using the notation in Section 2.4, row i of an order basis $\mathbf{M}(Z)$ of degree $\vec{\mu}$ and order $\vec{\omega}$ can be represented as the coefficient vector $\mathbf{M}_{\vec{\mu}-\vec{e}+\vec{e}_i}{}^{(i,\cdot)}$. As in (2.6), we have

$$\mathbf{M}_{\vec{\mu}-\vec{e}+\vec{e}_i}{}^{(i,\cdot)} \cdot K(\vec{\mu}-\vec{e}+\vec{e}_i, \vec{\omega}) = \mathbf{0}. \tag{3.9}$$

If an order basis of degree $\vec{\mu}$ and order $\vec{\omega}$ exists, then Theorem 3.13 implies that

$$\text{rank } K(\vec{\mu}-\vec{e}+\vec{e}_i, \vec{\omega}) = \text{rank } K(\vec{\mu}-\vec{e}, \vec{\omega}) = |\vec{\mu}|. \tag{3.10}$$

Thus, $K^*(\vec{\mu}-\vec{e}, \vec{\omega})$ is a $|\vec{\mu}| \times |\vec{\mu}|$ submatrix of $K(\vec{\mu}-\vec{e}, \vec{\omega})$, and we can rewrite (3.9) as

$$\mathbf{M}_{\vec{\mu}-\vec{e}}{}^{(i,\cdot)} \cdot K^*(\vec{\mu}-\vec{e}, \vec{\omega}) = -d \cdot b^*(\vec{\mu}, i), \tag{3.11}$$

where $b^*(\vec{\mu}, i)$ is the row of $K^*(\vec{\mu}-\vec{e}+\vec{e}_i, \vec{\omega})$ corresponding to $Z^{\vec{\mu}^{(i)}} \cdot \mathbf{F}(Z)^{(i)}$. By choosing $d = \pm \det K^*(\vec{\mu}-\vec{e}, \vec{\omega})$, we obtain a solution with entries in $\mathbb{D}$ by Cramer's rule. We give the corresponding chosen order basis a special name.

Definition 3.15 *We call d a* multi-gradient *if $d = \pm \det K^*(\vec{\mu} - \vec{e}, \vec{\omega})$ for some degree $\vec{\mu}$ and order $\vec{\omega}$. An order basis with a multi-gradient d in Definition 3.10(c) is called a* Mahler system. □

Moreover, we may formally write down a determinantal representation of the elements of a Mahler system. Namely,

$$\mathbf{M}(Z)^{(i,j)} = \pm \det \left[\; K^*(\vec{\mu} - \vec{e} + \vec{e}_i, \vec{\omega}) \; \middle| \; \mathbf{E}_{j, \vec{\mu}^{(j)} - 1 + \delta_{i,j}}(Z) \; \right] \tag{3.12}$$

with

$$\mathbf{E}_{j,\nu}(Z) = [0, \ldots, 0 | 1, Z, \ldots, Z^{\nu} | 0, \ldots, 0]^T ,$$

where the nonzero entries in $\mathbf{E}_{j,\nu}(Z)$ occurring in the jth stripe. In addition, we have

$$\begin{aligned} \mathbf{R}(Z)^{(i,j)} \cdot Z^{\vec{\omega}} &= \sum_{k=1}^{m} \mathbf{M}(Z)^{(i,k)} \cdot \mathbf{F}(Z)^{(k,j)} \\ &= \pm \det \left[\; K^*(\vec{\mu} - \vec{e} + \vec{e}_i, \vec{\omega}) \; \middle| \; \mathbf{E}_{j, \vec{\mu} - \vec{e} + \vec{e}_i}(Z) \; \right], \end{aligned} \tag{3.13}$$

where

$$\mathbf{E}_{j,\vec{\nu}}(Z) = \left[\mathbf{F}(Z)^{(1,j)}, \ldots, Z^{\vec{\nu}^{(1)} - 1} \cdot \mathbf{F}(Z)^{(1,j)} | \cdots | \mathbf{F}(Z)^{(m,j)}, \ldots, Z^{\vec{\nu}^{(m)} - 1} \cdot \mathbf{F}(Z)^{(m,j)} \right]^T .$$

In both (3.12) and (3.13) the matrices have commutative entries in all but the last column. It is understood that the determinant in both cases is expanded along the last column.

Example 3.16 *Let $a(Z), b(Z), \overline{a}(\hat{Z}), \overline{b}(\hat{Z})$ be the shift polynomials defined in Example 3.14. If $\vec{\mu} = (0, t+1)$ and $\vec{\omega} = (t+1)$, the system of equations (3.9) for the first row has the coefficient matrix*

$$K(\vec{\mu} - \vec{e} + \vec{e}_1, \vec{\omega}) = \begin{bmatrix} \overline{a}_0 & & & \cdots & & & \overline{a}_{d_a} \\ \overline{b}_0 & & \cdots & \overline{b}_{d_b} & & & \\ & \hat{\sigma}(\overline{b}_0) & & \cdots & \hat{\sigma}(\overline{b}_{d_b}) & & \\ & & \ddots & & & \ddots & \\ & & & \hat{\sigma}^t(\overline{b}_0) & & \cdots & \hat{\sigma}^t(\overline{b}_{d_b}) \end{bmatrix}. \tag{3.14}$$

By (3.6), we see that $K(\vec{\mu} - \vec{e} + \vec{e}_1, \vec{\omega})$ can also be written as

$$K(\vec{\mu} - \vec{e} + \vec{e}_1, \vec{\omega}) = \begin{bmatrix} a_{d_a} & & & \cdots & & & a_0 \\ \sigma^t(b_{d_b}) & & \cdots & \sigma^t(b_0) & & & \\ & \sigma^{t-1}(b_{d_b}) & & \cdots & \sigma^{t-1}(b_0) & & \\ & & \ddots & & & \ddots & \\ & & & b_{d_b} & & \cdots & b_0 \end{bmatrix}. \tag{3.15}$$

It is clear that rank $K(\vec{\mu} - \vec{e} + \vec{e}_1, \vec{\omega}) \geq t+1$ from the last $t+1$ rows. We see that fraction-free Gaussian elimination on $K(\vec{\mu} - \vec{e} + \vec{e}_1, \vec{\omega})$ using the rows of $b(Z)$ as pivots corresponds to the left pseudo-division of $a(Z)$ by $b(Z)$. The pseudo-remainder can also be obtained using (3.13). □

3.4 Fraction-free Recursion Formulas for Order Bases

In this section we show how to compute order bases in a fraction-free way recursively. This can also be thought of as constructing a sequence of eliminates of lower order terms of $\mathbf{F}(Z)$. In terms of linear algebra, the recursion can be viewed as a type of fraction-free Gaussian elimination which takes into consideration the special structure of the striped Krylov matrix associated to the elimination problem.

For a Mahler system $\mathbf{M}(Z)$ of degree $\vec{\mu}$ and order $\vec{\omega}$, we look at the terms of the residuals that we wish to eliminate. If they are all equal to zero then there is no need to eliminate and we already have an order basis of a higher order. Otherwise, we give recursive formulas for constructing an order basis of higher order and degree. However, the formulas involve divisions and a priori the new order basis may have coefficients in $\mathbb{Q}_{\mathbb{D}}$. In our case, the new order basis will be a Mahler system according to the existence and uniqueness results established before, and hence we will obtain the order bases and the residuals with coefficients in $\mathbb{D}$.

In the following theorem we give a recurrence relation which closely follows the case of shift polynomials [10] and the commutative case [12, Theorem 6.1(c)]. The resulting order bases have properties similar to those in [12, Theorems 7.2 and 7.3].

Theorem 3.17 *Let $\mathbf{M}(Z)$ be an order basis of degree $\vec{\mu}$ and order $\vec{\omega}$, and $\lambda \in \{1, \ldots, s\}$. Denote by $r_j = c_{\vec{\omega}^{(\lambda)}}\left((\mathbf{M}(Z) \cdot \mathbf{F}(Z))^{(j,\lambda)}\right)$, the (j, λ) entry of the first term of the residual of $\mathbf{M}(Z)$. Finally, set $\widetilde{\vec{\omega}} := \vec{\omega} + \vec{e}_\lambda$.*

(a) If $r_1 = \cdots = r_m = 0$ then $\widetilde{\mathbf{M}}(Z) := \mathbf{M}(Z)$ is an order basis of degree $\widetilde{\vec{\mu}} := \vec{\mu}$

and order $\widetilde{\vec{\omega}}$.

(b) *Otherwise, let π be an index such that $r_\pi \neq 0$. Then an order basis $\widetilde{\mathbf{M}}(Z)$ of degree $\widetilde{\vec{\mu}} := \vec{\mu} + \vec{e}_\pi$ and order $\widetilde{\vec{\omega}}$ with coefficients in $\mathbb{Q}_{\mathbb{D}}$ is obtained via the formulas*

$$p_\pi \cdot \widetilde{\mathbf{M}}(Z)^{(\ell,k)} = r_\pi \cdot \mathbf{M}(Z)^{(\ell,k)} - r_\ell \cdot \mathbf{M}(Z)^{(\pi,k)} \tag{3.16}$$

for $\ell, k = 1, \ldots, m$, $\ell \neq \pi$, and

$$\sigma(p_\pi) \cdot \widetilde{\mathbf{M}}(Z)^{(\pi,k)} = (r_\pi \cdot Z - \delta(r_\pi)) \cdot \mathbf{M}(Z)^{(\pi,k)} - \sum_{\ell \neq \pi} \sigma(p_\ell) \cdot \widetilde{\mathbf{M}}(Z)^{(\ell,k)} \tag{3.17}$$

for $k = 1, \ldots, m$, where $p_j = c_{\vec{\mu}^{(j)} + \delta_{\pi,j} - 1}\left(\mathbf{M}(Z)^{(\pi,j)}\right)$.

(c) *If in addition $\mathbf{M}(z)$ is a Mahler system of order $\vec{\omega}$ and degree $\vec{\mu}$, then $\widetilde{\mathbf{M}}(Z)$ is a Mahler system of order $\widetilde{\vec{\omega}}$ and degree $\widetilde{\vec{\mu}}$. In particular, $\widetilde{\mathbf{M}}(Z)$ has coefficients in $\mathbb{D}$.*

Proof. Part (a) is clear from the fact that the rows of $\mathbf{M}(Z)$ have order $\widetilde{\vec{\omega}}$ when $r_1 = \cdots = r_m = 0$.

For part (b) notice first that $\widetilde{\mathbf{M}}(Z)^{(\ell,\cdot)}$ for $\ell \neq \pi$ has order $\widetilde{\vec{\omega}}$ by construction, as required in Definition 3.10(a). In addition the row $(r_\pi \cdot Z - \delta(r_\pi)) \cdot \mathbf{M}(Z)^{(\pi,\cdot)}$ also has order $\widetilde{\vec{\omega}}$ since $(r_\pi \cdot Z - \delta(r_\pi))(r_\pi) = r_\pi \sigma(r_\pi) \cdot Z$. By construction therefore row $\widetilde{\mathbf{M}}(Z)^{\pi,\cdot}$ has order $\widetilde{\vec{\omega}}$.

We now focus on the properties of Definition 3.10(b). If $\mathbf{P}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$ has order $\widetilde{\vec{\omega}}$, then it has order $\vec{\omega}$ and so there exists $\mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$ such that

$$\mathbf{P}(Z) = \sum_{j=1}^{m} \mathbf{Q}(Z)^{(1,j)} \cdot \mathbf{M}(Z)^{(j,\cdot)}.$$

Applying (3.16) to rows $\ell \neq \pi$ results in

$$\mathbf{P}(Z) = \hat{\mathbf{Q}}(Z)^{(1,\pi)} \cdot \mathbf{M}(Z)^{(\pi,\cdot)} + \sum_{j\neq\pi} \hat{\mathbf{Q}}(Z)^{(1,j)} \cdot \widetilde{\mathbf{M}}(Z)^{(j,\cdot)} \tag{3.18}$$

where

$$\hat{\mathbf{Q}}(Z)^{(1,j)} = \begin{cases} \mathbf{Q}(Z)^{(1,j)} \cdot \frac{p_\pi}{r_\pi} & j \neq \pi \\ \sum_{i=1}^{m} \mathbf{Q}(Z)^{(1,i)} \cdot \frac{r_i}{r_\pi} & j = \pi \end{cases}. \tag{3.19}$$

Since $\mathbf{P}(Z)$ and all the $\widetilde{\mathbf{M}}(Z)^{(j,\cdot)}$ terms for $j \neq \pi$ have order $\widetilde{\vec{\omega}}$ this must also be the case for $\hat{\mathbf{Q}}(Z)^{(1,\pi)} \cdot \mathbf{M}(Z)^{(\pi,\cdot)}$. We can divide $\hat{\mathbf{Q}}(Z)^{(1,\pi)}$ by $r_\pi \cdot Z - \delta(r_\pi)$ on the right to obtain

$$\hat{\mathbf{Q}}(Z)^{(1,\pi)} = q(Z) \cdot (r_\pi \cdot Z - \delta(r_\pi)) + r$$

for some $q(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$ and $r \in \mathbb{Q}_{\mathbb{D}}$. Since $\hat{\mathbf{Q}}(Z)^{(1,\pi)} \cdot \mathbf{M}(Z)^{(\pi,\cdot)}$ has order $\widetilde{\vec{\omega}}$ and $(r_\pi \cdot Z - \delta(r_\pi))r_\pi = r_\pi\sigma(r_\pi)Z$, we see that $r \cdot r_\pi = 0$. Therefore, $r = 0$ by our choice of π. Hence,

$$\mathbf{P}(Z) = \sum_{j\neq\pi} \hat{\mathbf{Q}}(Z)^{(1,j)} \cdot \widetilde{\mathbf{M}}(Z)^{(j,\cdot)} + q(Z) \cdot (r_\pi \cdot Z - \delta(r_\pi)) \cdot \mathbf{M}(Z)^{\pi,\cdot}. \tag{3.20}$$

Completing the row operations which normalize the degrees of $\widetilde{\mathbf{M}}(Z)$ in (3.17) gives a $\widetilde{\mathbf{Q}}(Z)$ with $\mathbf{P}(Z) = \widetilde{\mathbf{Q}}(Z) \cdot \widetilde{\mathbf{M}}(Z)$. Consequently, the property of Definition 3.10(b) holds.

The verification of the new degree constraints of Definition 3.10(c) (with $\vec{\mu}$ being replaced by $\widetilde{\vec{\mu}}$) for $\widetilde{\mathbf{M}}(Z)$ is straightforward and is the same as in the commutative case [12, Theorem 7.2]. In addition, notice that p_π is the leading coefficient of $\mathbf{M}(Z)^{(\ell,\ell)}$, so the leading coefficient of $\widetilde{\mathbf{M}}(Z)^{(\ell,\ell)}$ equals r_π for all ℓ by construction. This shows part (b) of the theorem.

To show (c), we see from Section 3.3 and the existence of order bases of a specified degree and order that both $(\vec{\mu},\vec{\omega})$ and $(\widetilde{\vec{\mu}},\widetilde{\vec{\omega}})$ satisfy (3.10). By the uniqueness result of Theorem 3.13 we only need to show that the "leading coefficient" $\widetilde{d}$ of $\widetilde{\mathbf{M}}(Z)$ in Definition 3.10(c) is a multigradient of $(\widetilde{\vec{\mu}},\widetilde{\vec{\omega}})$, the latter implying that $\widetilde{\mathbf{M}}(Z)$ is a Mahler system and in particular has coefficients from $\mathbb{D}$.

Denote by d the corresponding "leading coefficient" of $\mathbf{M}(Z)$. In the case discussed in part (a), we do not increase the rank by going from $K(\vec{\mu},\vec{\omega})$ to $K(\widetilde{\vec{\mu}},\widetilde{\vec{\omega}})$ since we just add one column and keep full row rank. Hence $d=\widetilde{d}$ being a multigradient with respect to $(\vec{\mu},\vec{\omega})$ is also a multigradient with respect to $(\widetilde{\vec{\mu}},\widetilde{\vec{\omega}})$. In the final case described in part (b) we have $\widetilde{d}=r_\pi$. Using formula (3.13) for the residual of the πth row of $\mathbf{M}(Z)$ we see that r_π coincides (up to a sign) with the determinant of a submatrix of order $|\widetilde{\vec{\mu}}|$ of $K(\widetilde{\vec{\mu}},\widetilde{\vec{\omega}})$. Since $r_\pi \neq 0$ by construction, it follows that $\widetilde{d}=r_\pi$ is a new multigradient, as required for the conclusion. $\square$

In fact, we can make a stronger statement about the recurrence formulas (3.16) and (3.17) when $\mathbf{M}(Z)$ is a reduced order basis and the pivot π is chosen in a special way.

Corollary 3.18 *If $\mathbf{M}(Z)$ is a reduced order basis then the order basis $\widetilde{\mathbf{M}}(Z)$ computed by (3.16) and (3.17) in Theorem 3.17 is also a reduced order basis of degree*

$\widetilde{\vec{\mu}}$ and order $\widetilde{\vec{\omega}}$, provided that the pivot π is chosen such that

$$\vec{\mu}^{(\pi)} = \min_{1 \le j \le m} \left\{ \vec{\mu}^{(j)} : r_j \neq 0 \right\}. \tag{3.21}$$

Proof. First, note that if $\mathbf{M}(Z)$ is a reduced order basis, then $p_\ell = 0$ whenever $\vec{\mu}^{(\pi)} < \vec{\mu}^{(\ell)} - 1$, so that formula (3.17) can be rewritten as

$$\sigma(p_\pi) \cdot \widetilde{\mathbf{M}}(Z)^{(\pi,k)} = (r_\pi \cdot Z - \delta(r_\pi)) \cdot \mathbf{M}(Z)^{(\pi,k)} - \sum_{\vec{\mu}^{(\ell)} \le \vec{\mu}^{(\pi)}+1} \sigma(p_\ell) \cdot \widetilde{\mathbf{M}}(Z)^{(\ell,k)}, \tag{3.22}$$

showing that $\deg \widetilde{\mathbf{M}}(Z)^{(\pi,\cdot)} = \widetilde{\vec{\mu}}^{(\pi)}$. If $\ell \neq \pi$ then it is easy to see that the pivoting strategy (3.21) gives $\deg \widetilde{\mathbf{M}}(Z)^{(\ell,\cdot)} = \widetilde{\vec{\mu}}^{(\ell)}$. Thus, $\operatorname{rdeg} \widetilde{\mathbf{M}}(Z) = \widetilde{\vec{\mu}}$, and hence, by Lemma 3.3(a), it suffices to show that $\operatorname{cdeg} \widetilde{\mathbf{Q}}(Z) \le (\deg \mathbf{P}(Z)) \cdot \vec{e} - \widetilde{\vec{\mu}}$, with $\mathbf{P}(Z) = \widetilde{\mathbf{Q}}(Z) \cdot \widetilde{\mathbf{M}}(Z)$ as in the proof of Theorem 3.17.

By Lemma 3.3(a), we have $\operatorname{cdeg} \mathbf{Q}(Z) \le (\deg \mathbf{P}(Z)) \cdot \vec{e} - \vec{\mu}$ because $\mathbf{M}(Z)$ is a reduced order basis. We see in (3.19) that $\deg \hat{\mathbf{Q}}(Z)^{(1,j)} \le \deg \mathbf{P}(Z) - \vec{\mu}^{(j)} = \deg \mathbf{P}(Z) - \widetilde{\vec{\mu}}^{(j)}$ for all $j \neq \pi$ while $\deg \hat{\mathbf{Q}}(Z)^{(1,\pi)} \le \deg \mathbf{P}(Z) - \vec{\mu}^{(\pi)}$ because of the minimality of $\vec{\mu}^{(\pi)}$. In (3.20), $\deg q(Z) \le \deg \mathbf{P}(Z) - (\vec{\mu}^{(\pi)} + 1) = \deg \mathbf{P}(Z) - \widetilde{\vec{\mu}}^{(\pi)}$. Completing the row operations which normalize the degrees of $\widetilde{\mathbf{M}}(Z)$ in (3.22) gives $\widetilde{\mathbf{Q}}(Z)$ with $\mathbf{P}(Z) = \widetilde{\mathbf{Q}}(Z) \cdot \widetilde{\mathbf{M}}(Z)$ having the correct degree bounds. □

Once again, we will illustrate the recursion formulas with an example related to pseudo-division of shift polynomials.

Example 3.19 *Let $\overline{a}(\hat{Z})$ and $\overline{b}(\hat{Z})$ be those defined in Example 3.14. If we use the* *pivoting strategy*

$$\pi = \max_i \{i : r_i \neq 0\},$$

we find that $p_1 = 0$ *at each step, so we obtain* $\mathbf{M}(\hat{Z})$ *given in (3.8) with* $d = \pm \bar{b}_0^{[t+1]}$. *By Theorem 3.17,* $\mathbf{M}(\hat{Z})$ *is an order basis.* □

3.5 The FFReduce Algorithm

Theorem 3.17 gives a computational procedure that results in the FFReduce algorithm given in Algorithm 3.5. The resulting algorithm computes the rank and a row-reduced basis of the left nullspace $\mathcal{N}_{\mathbf{F}(Z)}$. For brevity, we will drop the indeterminate Z in the matrices of Ore polynomials in the description. Since we are interested in a fraction-free algorithm, we will assume that $\mathbf{F}(Z) \in \mathbb{D}[Z;\sigma,\delta]^{m\times s}$. This can be achieved by clearing the denominators of the entries in the matrix.

We now prove that the algorithm is correct.

Theorem 3.20 *Let* $r = \text{rank}\, \mathbf{F}(Z)$. *Then the final residual* $\mathbf{R}(Z)$ *computed by the* FFReduce *algorithm has rank* r *and* $m-r$ *zero rows. Moreover, if* $J \subseteq \{1,\ldots,m\}$ *is the set of row indices corresponding to the zero rows of* $\mathbf{R}(Z)$, *then the rows of* $\mathbf{M}(Z)^{(J,\cdot)}$ *form a row-reduced basis of the left nullspace* $\mathcal{N}_{\mathbf{F}(Z)}$.

Proof. We first recall that the last computed Mahler system $\mathbf{M}(Z)$ results from iteration $k = s\kappa$, $\kappa = mN + 1$, and has order $\vec{\omega} = \kappa \cdot \vec{e}$ and degree $\vec{\mu}$.

The statement rank $\mathbf{F}(Z) =$ rank $\mathbf{R}(Z)$ follows from Lemma 3.7 since $\mathbf{R}(Z) \cdot Z^{\kappa}$ is obtained from $\mathbf{F}(Z)$ by applying row operations of the first type.

Algorithm 3.1 The FFReduce Algorithm

Input: Matrix of Ore polynomials $\mathbf{F} \in \mathbb{D}[Z;\sigma,\delta]^{m\times s}$.

Output: Mahler system $\mathbf{M} \in \mathbb{D}[Z;\sigma,\delta]^{m\times m}$ of degree $\vec{\mu}$ and order $\vec{\omega}$, and residual $\mathbf{R} \in \mathbb{D}[Z;\sigma,\delta]^{m\times s}$.

```
{Initialization}
$\mathbf{M}_0 \leftarrow \mathbf{I}_m$, $\mathbf{R}_0 \leftarrow \mathbf{F}$, $d_0 \leftarrow 1$, $\vec{\mu}_0 \leftarrow \vec{0}$, $\vec{\omega}_0 \leftarrow \vec{0}$, $N \leftarrow \deg \mathbf{F}$, $\rho \leftarrow 0$, $k \leftarrow 0$

while $k < (mN+1)s$ do
  $\rho_k \leftarrow \rho$, $\rho \leftarrow 0$
  for all $\lambda = 1, \ldots, s$ do
    for all $\ell = 1, \ldots, m$ do
      $r_\ell \leftarrow c_0\left({\mathbf{R}_k}^{(\ell,\lambda)}\right)$                {first term of residuals}
    end for
    $\Lambda \leftarrow \{\ell \in \{1,\ldots,m\} : r_\ell \neq 0\}$
    if $\Lambda = \{\}$ then
      $\mathbf{M}_{k+1} \leftarrow \mathbf{M}_k$, $\mathbf{R}_{k+1} \leftarrow \mathbf{R}_k$, $d_{k+1} \leftarrow d_k$, $\vec{\mu}_{k+1} \leftarrow \vec{\mu}_k$
    else
      $\pi_k \leftarrow \min\left\{\ell \in \Lambda : \vec{\mu}_k^{(\ell)} = \min_j \left\{\vec{\mu}_k^{(j)} : j \in \Lambda\right\}\right\}$      {choose pivot}
      for all $\ell = 1, \ldots, m$, $\ell \neq \pi_k$ do
        $p_\ell \leftarrow c_{\vec{\mu}_k^{(\ell)}-1}\left({\mathbf{M}_k}^{(\pi_k,\ell)}\right)$
        ${\mathbf{M}_{k+1}}^{(\ell,\cdot)} \leftarrow \frac{1}{d_k}\left[r_{\pi_k} \cdot {\mathbf{M}_k}^{(\ell,\cdot)} - r_\ell \cdot {\mathbf{M}_k}^{(\pi_k,\cdot)}\right]$      {apply (3.16)}
        ${\mathbf{R}_{k+1}}^{(\ell,\cdot)} \leftarrow \frac{1}{d_k}\left[r_{\pi_k} \cdot {\mathbf{R}_k}^{(\ell,\cdot)} - r_\ell \cdot {\mathbf{R}_k}^{(\pi_k,\cdot)}\right]$
      end for

      {apply (3.17)}
      ${\mathbf{M}_{k+1}}^{(\pi_k,\cdot)} \leftarrow \frac{1}{\sigma(d_k)}\left[(r_{\pi_k} \cdot Z - \delta(r_{\pi_k})) \cdot {\mathbf{M}_k}^{(\pi_k,\cdot)} - \sum_{\ell\neq\pi_k} \sigma(p_\ell) \cdot {\mathbf{M}_{k+1}}^{(\ell,\cdot)}\right]$
      ${\mathbf{R}_{k+1}}^{(\pi_k,\cdot)} \leftarrow \frac{1}{\sigma(d_k)}\left[(r_{\pi_k} \cdot Z - \delta(r_{\pi_k})) \cdot {\mathbf{R}_k}^{(\pi_k,\cdot)} - \sum_{\ell\neq\pi_k} \sigma(p_\ell) \cdot {\mathbf{R}_{k+1}}^{(\ell,\cdot)}\right]$

      $d_{k+1} \leftarrow r_{\pi_k}$, $\vec{\mu}_{k+1} \leftarrow \vec{\mu}_k + \vec{e}_{\pi_k}$, $\rho \leftarrow \rho + 1$
    end if
    ${\mathbf{R}_{k+1}}^{(\ell,\lambda)} \leftarrow {\mathbf{R}_{k+1}}^{(\ell,\lambda)}/Z$ (formally)      {adjust residual in column $\lambda$}
    $\vec{\omega}_{k+1} \leftarrow \vec{\omega}_k + \vec{e}_\lambda$, $k \leftarrow k+1$
  end for
end while
$\mathbf{M} \leftarrow \mathbf{M}_k$, $\mathbf{R} \leftarrow \mathbf{R}_k$, $\vec{\mu} \leftarrow \vec{\mu}_k$, $\vec{\omega} \leftarrow \vec{\omega}_k$
```

In order to show that $\mathbf{R}(Z)$ has $m-r$ zero rows, let $\mathbf{W}(Z)$ be the row-reduced basis of $\mathcal{N}_{\mathbf{F}(Z)}$ as in Theorem 3.5, with $\vec{\alpha} = \text{rdeg}\, \mathbf{W}(Z)$ such that $\vec{\alpha} \le (m-1)N \cdot \vec{e}$. Since the rows of $\mathbf{W}(Z)$ have order $\kappa \cdot \vec{e}$, there exists $\mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{(m-r)\times m}$ such that $\mathbf{W}(Z) = \mathbf{Q}(Z) \cdot \mathbf{M}(Z)$. By construction and Corollary 3.18, $\mathbf{M}(Z)$ is a reduced order basis, and therefore row-reduced with row degree $\vec{\mu}$. Lemma 3.3(c) then implies that there is some permutation $p : \{1,\ldots,m-r\} \to \{1,\ldots,m\}$, with $\vec{\alpha}^{(j)} \ge \vec{\mu}^{(p(j))}$ for $j = 1,\ldots,m-r$. Hence, for $j = 1,\ldots,m-r$,

$$\begin{aligned}\deg \mathbf{R}(Z)^{(p(j),\cdot)} &= -\kappa + \deg\left(\mathbf{R}(Z)^{(p(j),\cdot)} \cdot Z^{\kappa\cdot\vec{e}}\right) = -\kappa + \deg\left(\mathbf{M}(Z)^{(p(j),\cdot)} \cdot \mathbf{F}(Z)\right)\\ &\le -\kappa + N + \deg \mathbf{M}(Z)^{(p(j),\cdot)} = -\kappa + N + \vec{\mu}^{(p(j))}\\ &\le -\kappa + N + \vec{\alpha}^{(j)} \le -\kappa + mN = -1,\end{aligned}$$

showing that these $m-r$ rows $\mathbf{R}(Z)^{(p(j),\cdot)}$ are indeed zero rows.

It remains to show that the rows of $\mathbf{M}(Z)^{(J,\cdot)}$ form a row-reduced basis of $\mathcal{N}_{\mathbf{F}(Z)}$. The submatrix $\mathbf{M}(Z)^{(J,\cdot)}$ is row-reduced because it consists of the rows of the row-reduced matrix $\mathbf{M}(Z)$. Since any $\mathbf{P}(Z) \in \mathcal{N}_{\mathbf{F}(Z)}$ has order $\kappa\cdot\vec{e}$, there exists $\mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]^{1\times m}$ such that $\mathbf{P}(Z) = \mathbf{Q}(Z)\cdot\mathbf{M}(Z)$. Thus,

$$\mathbf{Q}(Z)\cdot\mathbf{R}(Z)\cdot Z^{\kappa} = \mathbf{Q}(Z)\cdot\mathbf{M}(Z)\cdot\mathbf{F}(Z) = \mathbf{P}(Z)\cdot\mathbf{F}(Z) = \mathbf{0}.$$

The relation $r = \text{rank}\, \mathbf{R}(Z)$ implies that the nonzero rows of $\mathbf{R}(Z)$ are $\mathbb{Q}_{\mathbb{D}}[Z;\sigma,\delta]$-linearly independent, and hence $\mathbf{Q}(Z)^{(1,j)} = 0$ for $j \notin J$. Consequently, the rows of $\mathbf{M}(Z)^{(J,\cdot)}$ form a basis of $\mathcal{N}_{\mathbf{F}(Z)}$. $\square$

The theorem above was based on the estimate $\vec{\alpha}^{(j)} \le (m-1)N$ for the left minimal indices of $\mathcal{N}_{\mathbf{F}(Z)}$, which for general polynomial matrices is quite pessimistic, but can be attained, as shown in Remark 3.2. If a lower bound γ is available for $|\vec{\nu}|$ in Theorem 3.1, it would be sufficient to compute Mahler systems up to the final order $(mN + 1 - \gamma) \cdot \vec{e}$, since then we get from Theorem 3.1 and Theorem 3.5 the improved estimate $\vec{\alpha}^{(j)} \leq (m-1)N - \gamma$.

Remark 3.21 *The row-reduced basis of $\mathcal{N}_{\mathbf{F}(Z)}$ computed by the* FFREDUCE *algorithm is a scalar multiple of a matrix in Popov form by Remark 3.12.* □

3.6 Complexity of FFREDUCE

In this section, we examine the computational complexity of the FFREDUCE algorithm. We obtain bounds on the size of the intermediate results in the FFREDUCE algorithm, leading to a bound on the complexity of the algorithm. For our analysis, we assume that the coefficient domain $\mathbb{D}$ satisfies

$$\begin{aligned}
\text{size}(a+b) &= \mathcal{O}(\max(\text{size}(a), \text{size}(b))) \\
\text{size}(a \cdot b) &= \mathcal{O}(\text{size}(a) + \text{size}(b)) \\
\text{size}(\sigma(a)), \text{size}(\delta(a)) &= \mathcal{O}(\text{size}(a)) \\
\text{cost}(a+b) &= \mathcal{O}(\max(\text{size}(a), \text{size}(b))) \\
\text{cost}(a \cdot b) &= \mathcal{O}(\text{size}(a) \cdot \text{size}(b)) \\
\text{cost}(\sigma(a)) &= \mathcal{O}(\text{size}(a)^2) \\
\text{cost}(\delta(a)) &= \mathcal{O}(\text{size}(a)),
\end{aligned}$$

where the function "size" measures the total storage required for its arguments and the function "cost" estimates the number of bit operations required to perform the indicated arithmetic operations. These assumptions are valid in many cases. For example, if $\mathbb{D} = \mathbb{Z}$ or $\mathbb{D} = \mathbb{Z}[n]$, then size refers to the number of bits to represent an integer or the degree of the polynomial, respectively. The complexity analysis can easily be adapted for a different set of assumptions. For example, we may assume that the cost of multiplication is sub-quadratic, using fast multiplication algorithms such as Karatsuba's algorithm or the FFT method [42].

In what follows we denote by *cycle* the set of iterations $k = \kappa s, \kappa s + 1, \ldots, (\kappa + 1)s - 1$ in the FFReduce algorithm for some integer κ (that is, the execution of the inner loop). Let us first examine the size of the coefficients and the complexity of one iteration of algorithm FFReduce.

Lemma 3.22 *Let K be a bound on the size of the coefficients appearing in $\mathbf{F}(Z)^{(j,\cdot)}$, $Z \cdot \mathbf{F}(Z)^{(j,\cdot)}, \ldots, Z^{\vec{\mu}_k^{(j)}} \cdot \mathbf{F}(Z)^{(j,\cdot)}$ for $j = 1, \ldots, m$. Then the size of the coefficients in $\mathbf{M}_k$ and $\mathbf{R}_k$ is bounded by $\mathcal{O}(|\vec{\mu}_k|K)$. Moreover, the cost of iteration k is bounded by $\mathcal{O}((msN|\vec{\mu}_k|^2 + (m+s)|\vec{\mu}_k|^3)K^2)$.*

Proof. Equations (3.12) and (3.13) show that both the Mahler system and the residual can be represented as determinants of square matrices of order $|\vec{\mu}_k|$. The coefficients in these matrices are the coefficients of $\mathbf{F}(Z)^{(j,\cdot)}, Z \cdot \mathbf{F}(Z)^{(j,\cdot)}, \ldots, Z^{\vec{\mu}_k^{(j)}} \cdot \mathbf{F}(Z)^{(j,\cdot)}$. Hence the well-known Hadamard inequality [40] gives the above bound for the size of the coefficients.

In order to obtain the cost, we have to take into account the multiplication of each row of $(\mathbf{M}_k, \mathbf{R}_k)$ by two scalars and the multiplication of the pivot row by at most $m+1$ scalars. The number of additions is approximately the same and can be ignored. Also, there are m applications of σ and one application of δ. It remains to count the number of coefficients, and to take into account that each multiplication with a coefficient has a cost bounded by $\mathcal{O}(|\vec{\mu}_k|^2K^2)$. $\square$

By slightly generalizing [12, Theorem 6.2], we deduce the following complexity bound for the FFReduce algorithm.

Theorem 3.23 *Let K be a bound on the size of the coefficients appearing in $\mathbf{F}(Z)^{(j,\cdot)}, Z \cdot \mathbf{F}(Z)^{(j,\cdot)}, \ldots, Z^{\vec{\mu}_k^{(j)}} \cdot \mathbf{F}(Z)^{(j,\cdot)}$ for $j = 1, \ldots, m$, where $\vec{\mu}_k$ of iteration k of* FFReduce*. Then the total cost for computing $\mathbf{M}_k$ and $\mathbf{R}_k$ by the* FFReduce *algorithm is bounded by $\mathcal{O}((msN|\vec{\mu}_k|^3 + (m+s)|\vec{\mu}_k|^4)K^2)$. Therefore, the worst case bit complexity of the* FFReduce *algorithm is $\mathcal{O}((m+s)m^4s^4N^4K^2)$.*

Proof. The first part of the Theorem is an immediate consequence of Lemma 3.22 and of the fact that the number of iterations in the FFReduce algorithm in which any reduction is done equals $|\vec{\mu}_k|$. In order to show the second part, we use the fact that $|\vec{\mu}| \leq |\vec{\omega}|$ with $\vec{\omega} = (mN+1) \cdot \vec{e}$, and $|\vec{\omega}| = (mN+1)s$. $\square$

Remark 3.24 *If we assume that multiplication can be done in $\mathcal{O}\tilde{}(size(a)+size(b))$ operations*[1]*, the complexity in Theorem 3.23 becomes $\mathcal{O}\tilde{}((m+s)m^3s^3N^3K)$.* $\square$

Remark 3.25 *Suppose that $\mathbb{D}$ is the polynomial domain $\mathbb{Z}[x]$. We wish to consider the size of coefficients in terms of both the degree in x and the integer coefficients of x. For $a \in \mathbb{Z}[x]$, let $\deg_x(a)$ denote the degree of a with respect to x, and $\|a\|$ be*

[1]This is the "soft-O" notation. If $f(n) \in \mathcal{O}\tilde{}(g(n))$ then $f(n) \in \mathcal{O}(g(n)\log^b(n))$ for some $b \geq 0$.

the maximal absolute value of the integer coefficients of a*. Suppose that* $a, b \in \mathbb{Z}[x]$*. The model of arithmetic in* $\mathbb{Z}[x]$ *satisfies:*

$$\begin{aligned}
\mathit{size}(a) &= \mathcal{O}(\deg_x(a) \log \|a\|) \\
\mathit{size}(b) &= \mathcal{O}(\deg_x(b) \log \|b\|) \\
\mathit{size}(a+b) &= \mathcal{O}(\max(\mathit{size}(a), \mathit{size}(b))) \\
\mathit{size}(a \cdot b) &= \mathcal{O}\tilde{}(\mathit{size}(a) + \mathit{size}(b)) \\
\mathit{cost}(a+b) &= \mathcal{O}(\max(\mathit{size}(a), \mathit{size}(b))) \\
\mathit{cost}(a \cdot b) &= \mathcal{O}(\mathit{size}(a) \cdot \mathit{size}(b)).
\end{aligned}$$

Thus, the size of the intermediate results given in Lemma 3.22 still holds.

In the case where $\sigma(x) = \alpha x$ *for some nonzero* $\alpha \in \mathbb{Z}$ *and* $\delta(a) = 0$ *for all* $a \in \mathbb{Z}[x]$*, it can be shown that*

$$\begin{aligned}
\mathit{size}(Z^k \cdot a) &= \mathcal{O}(k\mathit{size}(a)^2 \log \alpha) \\
\mathit{cost}(Z^k \cdot a) &= \mathcal{O}(k\mathit{size}(a)^2 \log \alpha)
\end{aligned}$$

Therefore, we can set the quantity K *to be the maximum of* $((mN+1)s)\mathit{size}(a)^2 \log \alpha$ *where* a *ranges over all coefficients in* $\mathbf{F}(Z)$*. Performing the same analysis, we obtain a complexity of* $\mathcal{O}\tilde{}((m+s)m^4 s^4 N^4 K^2 \log \alpha)$*.*

Consider the case where $\sigma(x) = \alpha x + \beta$ *for some nonzero* $\alpha, \beta \in \mathbb{Z}$ *with* $\delta(a) = 0$

for all $a \in \mathbb{Z}[x]$*. Then*

$$\begin{aligned} size(Z^k \cdot a) &= \mathcal{O}(k\,size(a)^2 \log \alpha\beta) \\ cost(Z^k \cdot a)) &= \mathcal{O}(k\,size(a)^2 \log \alpha\beta) \end{aligned}$$

Therefore, we can set the quantity K *to be the maximum of* $((mN+1)s)size(a)\log\alpha$ *where* a *ranges over all coefficients in* $\mathbf{F}(Z)$*. Performing the same analysis, we obtain a complexity of* $\mathcal{O}\tilde{}((m+s)m^4s^4N^4K^2\log\alpha\beta)$*.*

Finally, we consider the differential case in which σ *is the identity and* $\delta(a) = \frac{d}{dx}a$ *for all* $a \in \mathbb{Z}[x]$*. Then*

$$\begin{aligned} size(Z \cdot a) &= \mathcal{O}(size(a) + \log size(a)) \\ cost(Z \cdot a) &= \mathcal{O}\tilde{}(size(a)) \end{aligned}$$

We can set the quantity K *to be the maximum of* $size(a) + (mN+1)s \log size(a)$ *over all coefficients* a *of* $\mathbf{F}(Z)$*. We obtain a complexity analysis of* $\mathcal{O}\tilde{}((m+s)m^4s^4N^4K^2)$*.*

A tighter estimate could be obtained if we specify the size and cost of the sums and products in two components ($\deg_x(a)$ *and* $\|a\|$*) separately [46]. We use the above model to simplify the presentation.*

Chapter 4

Fraction-free Algorithms for Matrices of Shift Polynomials

In this chapter we show how the FFReduce algorithm given in the previous chapter can be used to solve a number of different problems for $\mathbf{F}(Z) \in \mathbb{D}[Z;\sigma]^{m\times s}$. Of course, when σ is the identity function on $\mathbb{Q}_{\mathbb{D}}$ the same techniques give fraction-free algorithms for polynomial matrices. We will consider the computation of full rank decomposition (for finding solutions of linear functional systems), row-reduced form, and weak Popov form of a matrix of shift polynomial. We also show that our algorithm can be used to compute a GCRD and an LCLM of matrices of shift polynomials. Finally, we show that our algorithm can be used to compute subresultants of two shift polynomials [44, 45].

4.1 Additional Properties of FFReduce

In this section, we show a number of additional properties satisfied by the FFReduce algorithm when the input is restricted to matrices of shift polynomials. These properties are crucial in the development of the algorithms in this chapter.

We first show that any order basis has a "shifted left inverse," which is a variation of the notion of invertibility that will be useful in applications.

Lemma 4.1 *Let $\kappa \geq 0$, $\vec{\omega} = \kappa \cdot \vec{e} \in \mathbb{Z}^{1\times s}$, $\vec{\nu} = \kappa \cdot \vec{e} \in \mathbb{Z}^{1\times m}$. If $\mathbf{M}(Z)$ is an order basis of order $\vec{\omega}$ and degree $\vec{\mu}$, then there exists a* shifted left inverse $\mathbf{M}^*(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma]^{m\times m}$ *such that $\mathbf{M}^*(Z) \cdot \mathbf{M}(Z) = Z^{\vec{\nu}}$. Moreover, if $\mathbf{M}(Z)$ is a reduced order basis, then $\mathbf{M}^*(Z)$ satisfy the additional degree constraint cdeg $\mathbf{M}^*(Z) \leq \vec{\nu} - \vec{\mu}$.*

Proof. Every row of the matrix $Z^{\vec{\nu}}$ has order $\vec{\omega}$. Therefore, the existence of $\mathbf{M}^*(Z)$ is implied by Definition 3.10(b). If $\mathbf{M}(Z)$ is a reduced order basis then the degree constraint is implied by Remark 3.11. □

Next, we prove two lemmas relating the pivots used in each cycle to the rank of the trailing coefficient. This will be used in computing a row-reduced form and a weak Popov form of a matrix of shift polynomials.

Lemma 4.2 *Let $\kappa \geq 0$. Then $\rho_{(\kappa+1)s}$ = rank $\mathbf{R}_{\kappa s}(0)$. Furthermore, if $\mathbf{T}(Z)$ is the matrix formed by the rows of $\mathbf{R}_k(Z)$ chosen as pivots during the κth cycle, then $\mathbf{T}(0)$ is a matrix of full row rank $\rho_{(\kappa+1)s}$ and is in upper echelon form (up to row permutations).*

Proof. Denote by $H_k \in \mathbb{D}^{m\times s}$, the coefficient of Z^κ of $\mathbf{M}_k(Z) \cdot \mathbf{F}(Z)$, for $\kappa s \leq k \leq (\kappa+1)s$. First, we claim that when row π_k is chosen as a pivot for column

$k - \kappa s + 1$, the subspace generated by the rows of H_k is the same as the subspace generated by row π_k of H_k (a pivot row) and the rows of H_{k+1}. This is clearly true after the order has been increased for rows $\ell \neq \pi_k$ as the recurrence (3.16) is invertible. Multiplying row π_k of $\mathbf{M}_k(Z)$ by Z produces zeros in row π_k in the updated matrix, so that row π_k of H_k must be kept. Finally, the degree adjustment from rows $\ell \neq \pi_k$ is again invertible. Therefore, the subspaces are the same.

It follows that the rows of $H_0 = \mathbf{R}_{\kappa s}(0)$ span the same space as all pivot rows together with the rows of $H_{(\kappa+1)s}$. Since $\mathbf{M}_k(Z)$ is an order basis of order $\vec{\omega}_k \geq \vec{\omega}_{\kappa s} = \kappa \cdot \vec{e}$, it follows that the first $k - \kappa s$ columns of H_k are zero. Thus, $H_{(\kappa+1)s} = 0$. In addition, the $(k - \kappa s)$th component of the pivot row in iteration k equals $r_{\pi_k} \neq 0$. Therefore the pivot rows form a full row rank upper echelon matrix (up to row permutations). Since $\rho_{(\kappa+1)s}$ gives the number of pivot rows in the κth cycle, it follows that $\rho_{(\kappa+1)s} = \operatorname{rank} \mathbf{R}_{\kappa s}(0)$. □

Lemma 4.3 *The pivots used in one cycle of* FFReduce *are distinct, or equivalently, $\vec{\mu}_{(\kappa+1)s} \leq \vec{\mu}_{\kappa s} + \vec{e}$ for any $\kappa \geq 0$. Moreover, rank $\mathbf{R}_{\kappa s}(0)$ is increasing in κ.*

Proof. By the argument in Remark 3.12, we may assume that $\mathrm{LC}_{row}\left(\mathbf{M}_{\kappa s}(Z)\right)$ and $\mathrm{LC}_{row}\left(\mathbf{M}_{(\kappa+1)s}(Z)\right)$ are nonsingular and in upper echelon form after row permutation. By Remark 3.11, there exists $\mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma]^{m\times m}$ such that

$$Z \cdot \mathbf{M}_{\kappa s}(Z) = \mathbf{Q}(Z) \cdot \mathbf{M}_{(\kappa+1)s}(Z), \quad \deg \mathbf{Q}(Z)^{(j,\ell)} \leq \vec{\mu}_{\kappa s}^{(j)} + 1 - \vec{\mu}_{(\kappa+1)s}^{(\ell)} \text{ for all } j, \ell.$$

Comparing the coefficients at $Z^{\vec{\mu}_{\kappa s}^{(j)}+1}$ in position (j, ℓ), we have on the left a nonsingular upper triangular matrix, and on the right the leading row coefficient matrix

B of $\mathbf{Q}(Z)$ (with coefficients at power $\vec{\mu}_{\kappa s}^{(j)} + 1 - \vec{\mu}_{(\kappa+1)s}^{(\ell)}$) multiplied by an upper triangular matrix A. Since the entries of the coefficient matrices are in $\mathbb{Q}_{\mathbb{D}}$, A must be nonsingular, and so B is also nonsingular and hence upper triangular. Hence the degrees on the diagonal cannot be smaller than 0, showing that $\vec{\mu}_{\kappa s}^{(j)} + 1 \geq \vec{\mu}_{(\kappa+1)s}^{(j)}$, or, in other words, $\vec{\mu}_{(\kappa+1)s} \leq \vec{\mu}_{\kappa s} + \vec{e}$. Thus, the pivots in one cycle are distinct. Also, denoting by C the trailing coefficient of $\mathbf{Q}(Z)$, we easily obtain that $C \cdot \mathbf{R}_{(\kappa+1)s}(0)$ coincides with $\sigma(\mathbf{R}_{\kappa s}(0))$ (which has the same rank as $\mathbf{R}_{\kappa s}(0)$). Hence the rank of $\mathbf{R}_{\kappa s}(0)$ is increasing. □

The two lemmas above also allow us to terminate the algorithm earlier in most cases. In particular, we may terminate the algorithm when

$$\rho_{\kappa s} + \text{ number of zero rows in } \mathbf{R}_{\kappa s}(Z) \ = m. \tag{4.1}$$

For the remainder of this chapter we will assume that the FFReduce algorithm has been modified to use the termination condition (4.1).

Theorem 4.4 *The matrix* $\mathbf{R}(Z)$ *computed by* FFReduce *satisfies*

$$\textit{rank}\ \mathbf{R}(0) = \textit{rank}\ \mathbf{R}(Z) = \textit{rank}\ \mathbf{F}(Z).$$

Proof. By Theorem 3.20, rank $\mathbf{R}(Z)$ = rank $\mathbf{F}(Z)$. Furthermore, rank $\mathbf{R}_{\kappa s}(Z) =$ rank $\mathbf{F}(Z)$ for all κ by Lemma 3.7. If the algorithm stops after the $(\kappa+1)$st cycle and $r = \rho_{(\kappa+1)s}$, then

$$r = \text{rank } \mathbf{R}_{\kappa s}(0) \leq \text{rank } \mathbf{R}_{(\kappa+1)s}(0) \leq \text{rank } \mathbf{R}_{(\kappa+1)s}(Z) \leq r$$

by Lemma 4.2, Lemma 4.3, and the fact that $\mathbf{R}_{(\kappa+1)s}(Z)$ contains r nonzero rows. Consequently, $r = \operatorname{rank} \mathbf{R}_{(\kappa+1)s}(Z) = \operatorname{rank} \mathbf{F}(Z)$. □

Remark 4.5 *We have shown implicitly in the proof that, if we stop after cycle $(\kappa+1)$, then $\mathbf{R}_{\kappa s}(0)$ already has full rank $r = \rho_{(\kappa+1)s}$. Since all the pivots in one cycle are distinct by Lemma 4.3, it follows that the set of pivot rows in $\mathbf{R}_{\kappa s}(0)$ also has rank r.* □

We now give a tighter complexity bound of FFReduce in the case of matrices of shift polynomials.

Theorem 4.6 *Let K be an upper bound on the size of the coefficients appearing in $\mathbf{F}(Z)$. Then the total cost for computing $\mathbf{M}$ and $\mathbf{R}$ by the* FFReduce *algorithm is bounded by $\mathcal{O}((m+s)m^4 \min(m,s)^4 N^4 K^2)$.*

Proof. Since the pivots in one cycle are distinct by Lemma 4.3, the number of pivots in each cycle is bounded by $\operatorname{rank} \mathbf{F}(Z) \leq \min(m,s)$. This gives the bound $|\vec{\mu}| \leq \min(m,s)(mN+1)$. The complexity now follows from Theorem 3.23. □

4.2 Full Rank Decomposition and Solutions of Linear Functional Systems

When $\mathbf{F}(Z) \in \mathbb{D}[Z;\sigma]^{m\times s}$ represents a system of linear recurrence equations, one can show that an equivalent system with a nonsingular leading (or trailing) coefficient allows one to obtain bounds on the degrees of the numerator and the denominator of all rational solutions. This has been used to compute rational solutions of linear functional systems [1, 3, 4], which include systems defined by differential operators, difference operators, and q-difference operators. These systems are transformed into a system of linear recurrence equations by studying the action of the operators on the coefficients corresponding to an appropriately chosen basis (e.g. $\{x^n\}_{n\geq 0}$) as illustrated in Example 1.6. The transformation may introduce negative powers of Z, but they can be removed by multiplying by an appropriate power of Z on the right.

Let $\mathbb{Q}_{\mathbb{D}}[Z;\sigma][Z^{-1};\sigma^{-1}]$ be the iterated domain where we have the identities

$$Z \cdot Z^{-1} = Z^{-1} \cdot Z = 1, \quad Z \cdot a \cdot Z^{-1} = \sigma(a), \quad Z^{-1} \cdot a \cdot Z = \sigma^{-1}(a)$$

for all $a \in \mathbb{Q}_{\mathbb{D}}$. The rank revealing transformations of Abramov and Bronstein [3, 4] can be formalized as follows. Given $\mathbf{F}(Z) \in \mathbb{D}[Z;\sigma]^{m\times s}$, we wish to find $\mathbf{T}(Z^{-1}) \in \mathbb{D}[Z^{-1};\sigma^{-1}]^{m\times m}$ such that

$$\mathbf{T}(Z^{-1}) \cdot \mathbf{F}(Z) = \mathbf{W}(Z) \in \mathbb{D}[Z;\sigma]^{m\times s}, \tag{4.2}$$

with the number of nonzero rows r of $\mathbf{W}(Z)$ coinciding with the rank of the trailing coefficient W_0, and hence with the rank of $\mathbf{W}(Z)$. In addition we require the existence of $\mathbf{S}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma]^{m\times m}$ such that

$$\mathbf{S}(Z) \cdot \mathbf{T}(Z^{-1}) = \mathbf{I}_m.$$

Thus, the process of elimination for obtaining $\mathbf{W}(Z)$ is invertible. More precisely, we obtain for $\mathbf{F}(Z)$ the *full rank decomposition*

$$\mathbf{F}(Z) = \mathbf{S}(Z) \cdot \mathbf{W}(Z) = \widetilde{\mathbf{S}}(Z) \cdot \widetilde{\mathbf{W}}(Z) \tag{4.3}$$

with $\widetilde{\mathbf{W}}(Z) \in \mathbb{D}[Z;\sigma]^{r\times n}$ obtained by extracting the nonzero rows of $\mathbf{W}(Z)$, and $\widetilde{\mathbf{S}}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma]^{m\times r}$ by extracting from $\mathbf{S}(Z)$ the corresponding columns. Moreover, the rank of the trailing coefficient of $\widetilde{\mathbf{W}}(Z)$ is of full row rank r, and this quantity coincides with the rank of $\widetilde{\mathbf{W}}(Z)$. Finally, from (4.3) we see that the rank of $\mathbf{F}(Z)$ is bounded above by r, whereas (4.2) implies that $\operatorname{rank} \mathbf{F}(Z) \geq \operatorname{rank} \mathbf{W}(Z) = r$. Thus we have $r = \operatorname{rank} \mathbf{F}(Z)$.

The full rank decomposition problem can be solved by the FFReduce algorithm as follows.

Theorem 4.7 *Let $\mathbf{M}(Z)$ be the final order basis of order $\vec{\omega} = \kappa \cdot \vec{e}$ and degree $\vec{\mu}$, and let $\mathbf{M}^*(Z)$ be the shifted left inverse of $\mathbf{M}(Z)$ in Lemma 4.1. Then*

$$\begin{aligned}
\mathbf{W}(Z) &= Z^{-\kappa\cdot\vec{e}} \cdot \mathbf{R}(Z) \cdot Z^{\kappa\cdot\vec{e}} \\
\mathbf{T}(Z^{-1}) &= Z^{-\kappa\cdot\vec{e}} \cdot \mathbf{M}(Z) \\
\mathbf{S}(Z) &= Z^{-\kappa\cdot\vec{e}} \cdot \mathbf{M}^*(Z) \cdot Z^{\kappa\cdot\vec{e}}
\end{aligned}$$

solves the full rank decomposition problem (4.3).

Proof. The equations (4.2) and (4.3) can easily be verified by substitution, using the properties that $\mathbf{M}(Z) \cdot \mathbf{F}(Z) = \mathbf{R}(Z) \cdot Z^{\kappa\cdot\vec{e}}$ and $\mathbf{M}^*(Z) \cdot \mathbf{M}(Z) = Z^{\kappa\cdot\vec{e}}$. □

Remark 4.8 *We may modify the* FFReduce *algorithm to also compute the shifted* *inverse* $\mathbf{M}^*(Z)$ *incrementally by examining how* $\mathbf{Q}(Z)$ *is updated in the proof of Theorem 3.17. This corresponds to applying column operations to* $\mathbf{M}^*(Z)$ *to obtain the new shifted inverse. However, the entries of* $\mathbf{M}^*(Z)$ *may contain fractions and we no longer have a fraction-free algorithm. In practice, we only need to ensure that a shifted inverse exists. It is not important to compute it explicitly.* □

Example 4.9 *Let* $\mathbb{D} = \mathbb{Z}[n, 2^n]$ *and consider*

$$\mathbf{F}(Z) = \begin{bmatrix} 0 & -1 \\ 0 & -12356 \end{bmatrix} + \begin{bmatrix} -80 & 0 \\ -988480 & -8029 \end{bmatrix} Z + \begin{bmatrix} -32 & 0 \\ -1037712 & 750 \end{bmatrix} Z^2 + \begin{bmatrix} 0 & 0 \\ -196928 & -300 \end{bmatrix} Z^3 + \begin{bmatrix} 0 & 1 \\ 0 & 120 \end{bmatrix} Z^4 + \begin{bmatrix} 2^n(n+1) & 0 \\ 0 & 3077(n+1) \end{bmatrix} Z^5,$$

which is the same as the example from [3] except that it is multiplied (on the right) by Z^4. *Using our algorithm, we terminate at* $\vec{\omega} = (6, 6)$ *in which the residuals are not all zero in the last two iterations. The trailing coefficient of the residual* $\mathbf{R}(Z)$ *obtained one cycle earlier at* $\vec{\omega} = (5, 5)$ *has a determinant that is an integer constant times* $2^n(n + 1) - 80$.

Writing $\mathbf{W}(Z) = Z^{-(5,5)} \cdot \mathbf{R}(Z) \cdot Z^{(5,5)}$, *the determinant of the trailing coefficient of* $\mathbf{W}(Z)$ *is the same as that in [3], up to a constant. We remark that the product of all the factors removed during the complete process is*

$$235015917188033446164000000000\,(2^n(n + 1) - 80)\,.$$

□

4.2.1 Experimental Results

An exact arithmetic method involving coefficient GCD computations for the computation of $\mathbf{T}(Z^{-1}) \cdot \mathbf{F}(Z) = \mathbf{W}(Z)$ with $\mathbf{W}(Z)$ as above has already been given by Abramov and Bronstein [3, 4]. We now give some experimental results comparing their approach with FFReduce[1].

As we have mentioned in Section 2.5.1, their "fraction-free" algorithm cannot remove common divisors between two cycles (after the row degree is decreased) without coefficient GCD computations. In particular, for 2×2 matrices such as the one in Example 4.9, the algorithm always encounters a trailing coefficient of rank at most 1 except in the last cycle. In such a case fraction-free Gaussian elimination on the trailing coefficient alone does not remove any common factor at all. Unlike our algorithm, however, the pivot row in their algorithm is not modified and does not grow.

We compare FFReduce against Abramov and Bronstein's algorithm implemented as `LinearFunctionalSystems[MatrixTriangularization]` (we abbreviate this as AB) in Maple 8[2]. In this implementation, GCD computations are performed when computing the elements in the kernel of the trailing coefficient in order to obtain "small" vectors. No other GCD computations are performed.

Experiments generated randomly by applying random row operations in reverse

[1]The experiments were performed on an Intel Pentium 4 1.7 GHz machine with 1GB of RAM.

[2]This implementation performs additional optimizations when the trailing coefficient has a zero row or a zero column. This reduces the number of iterations required to obtain the final result. The FFReduce algorithm can be adopted to perform such shifts as well. In our comparison, such optimizations are enabled in AB but disabled in FFReduce. Disabling the optimizations in AB does not significantly affect the results in our experiments.

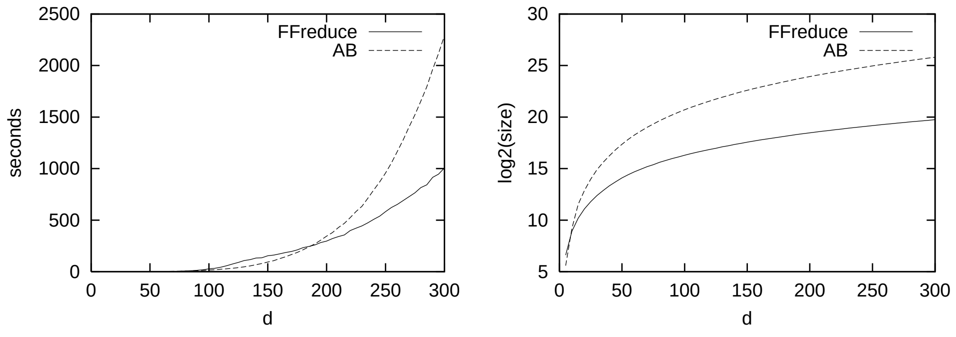
from a final desired result show that AB can be significantly faster than FFReduce. In all such cases, the reason is that the small vectors computed in the kernel effectively control the growth of coefficients. On the other hand, coefficient growth in FFReduce is larger even though it is controlled. In some cases, FFReduce can be slower by a large factor (e.g. 1000). When computing small vectors does not effectively control the coefficient growth, however, FFReduce is faster than AB.

We can construct examples in which controlling coefficient growth only in the kernel computation is insufficient. For the first set of experiments, we consider the commutative case in $\mathbb{Z}[z]^{2\times 2}$, where the input matrices are

$$\mathbf{F}_d(z) = \begin{bmatrix} \sum_{i=0}^{d} p_{i+1} z^i & \sum_{i=0}^{d-1} p_{i+1} z^i \\ \sum_{i=0}^{d} p_{i+d+2} z^i & \sum_{i=0}^{d-1} p_{i+d+2} z^i \end{bmatrix}, \tag{4.4}$$

where p_i is the ith prime. The choice of distinct primes as coefficients improves the chance that the "small" kernel vectors have size close to the ones obtained by fraction-free Gaussian elimination. Also, the number of iterations required is relatively large by this choice of the input. Figure 4.1 shows the computing time and the size of the final results (the length[3] of all integer coefficients) depending on d. We remark that the size of the final results for FFReduce includes both the residual and the transformation matrix, while the size for AB includes only the residual. As we expect, the growth in both the size and the computation time in AB is much higher than that in FFReduce. Note that in AB, the growth in the size of the intermediate results between cycles cannot be avoided regardless of the

[3]This is the result returned by the `length` command in Maple.



(a) Computation time (b) $\log_2$(size) of final results

Figure 4.1: Comparison of AB and FFReduce on input matrices defined in (4.4).

algorithm used to compute the kernel.

In the next set of experiments, we consider $\mathbf{F}(Z) \in \mathbb{Z}[n][Z;\sigma]^{2\times 2}$, where $\sigma(a(n)) = a(n+1)$. The input matrices are chosen to be

$$\mathbf{F}_d(Z) = \begin{bmatrix} q_{0,d}(Z) & q_{0,d-1}(Z) \\ q_{2d+2,d}(Z) & q_{2d+2,d-1}(Z) \end{bmatrix}, \tag{4.5}$$

where $q_{j,k}(Z) = \sum_{i=0}^{k}(p_{2i+j+1}n + p_{2i+j+2})Z^i$. Table 4.1 shows the experimental results. We see again that FFReduce performs much better than AB in this case. We also note that the growth in the size of the final results in FFReduce is much less than that in AB.

Next, we examine a set of experiments performed in $\mathbb{Z}[n][Z;\sigma]^{3\times 3}$ such that

	AB		FFReduce	
d	Time (sec)	Size	Time (sec)	Size
1	0.055	75	0.013	107
2	0.063	403	0.021	337
3	0.131	1015	0.035	689
4	0.330	3533	0.056	1327
5	0.651	10796	0.123	2092
6	1.541	35920	0.129	3302
7	3.949	101031	0.276	4684
8	13.994	288935	0.375	6743
9	145.125	719619	0.492	9008
10	720.742	1835504	0.730	12389
11	4246.829	4414659	1.329	15757
12	15025.581	10593921	2.442	20595

Table 4.1: Comparison of AB and FFReduce on input matrices defined in (4.5).

$\sigma(a(n)) = a(n+1)$. We define

$$\mathbf{F}_d(Z) = \begin{bmatrix} q_{0,d}(Z) & q_{0,d-1}(Z) & q_{0,d-2}(Z) \\ q_{2d+2,d}(Z) & q_{2d+2,d-1}(Z) & q_{4d+4,d-2}(Z) \\ q_{4d+4,d}(Z) & q_{4d+4,d-1}(Z) & q_{2d+2,d-2}(Z) \end{bmatrix}. \tag{4.6}$$

Both AB and FFReduce were applied on this set of input matrices, with a time limit of four hours. Table 4.2 shows the results, showing once again that FFReduce performs significantly better when coefficient growth cannot be avoided by simply controlling the size of the vectors in the kernel computation.

Finally, we give results on another class of examples in $\mathbb{Z}[n][Z;\sigma]^{2\times 2}$ with

	AB		FFReduce	
d	Time (sec)	Size	Time (sec)	Size
2	0.123	654	0.101	1488
3	0.125	2606	0.239	4589
4	0.287	7920	0.455	8621
5	0.691	27972	0.900	17267
6	1.582	84523	1.867	27208
7	4.656	265003	2.717	44369
8	19.342	714330	6.334	62900
9	331.509	1948947	20.334	92194
10	1943.193	4770766	148.652	122964
11	5821.765	12177824	516.682	169323
12	10144.400	27971967	631.781	213626
13	?	?	1528.602	280124
14	?	?	1660.289	340995
15	?	?	2403.154	432665

Table 4.2: Comparison of AB and FFReduce on input matrices defined in (4.6). An entry of "?" means that the test did not finish within four hours.

	AB		FFREDUCE	
d	Time (sec)	Size	Time (sec)	Size
1	0.422	1325	0.048	1430
2	2.547	12012	0.287	6057
3	47.703	123110	1.394	15706
4	2045.412	837021	5.059	26665
5	?	?	87.209	65876
6	?	?	228.244	102301
7	?	?	488.750	165810
8	?	?	924.529	229896
9	?	?	1417.518	296447
10	?	?	4439.551	453021

Table 4.3: Comparison of AB and FFREDUCE on input matrices defined in (4.7). An entry of "?" means that the test did not finish within three hours.

$\sigma(a(n)) = a(n+1)$. Let

$$\mathbf{F}_d(Z) = \begin{bmatrix} a_d(Z) \\ b_d(Z) \end{bmatrix} \cdot \begin{bmatrix} c_d(Z) & d_d(Z) \end{bmatrix}, \tag{4.7}$$

where $a_d(Z), b_d(Z), c_d(Z), d_d(Z)$ are random polynomials of degree d. Note that $\deg \mathbf{F}(Z) = 2d$ and rank $\mathbf{F}(Z) = 1$. The results are given in Table 4.3. Experiments on 3×3 matrices give similar results.

4.3 Computing a Row-reduced Form

The FFREDUCE algorithm can be used to compute a row-reduced form in the case of matrices of shift polynomials. In particular, given $\mathbf{F}(Z) \in \mathbb{D}[Z;\sigma]^{m\times s}$ we can compute $\mathbf{U}(Z)$ and $\mathbf{T}(Z)$ such that $\mathbf{U}(Z) \cdot \mathbf{F}(Z) = \mathbf{T}(Z)$ with the nonzero rows of $\mathbf{T}(Z)$ being row-reduced. Since we wish to eliminate high-order coefficients, we perform the substitution $\hat{Z} = Z^{-1}$, $\hat{\sigma} = \sigma^{-1}$ and perform the reduction over $\mathbb{D}[\hat{Z}; \hat{\sigma}]$. We further assume that σ^{-1} does not introduce fractions, so that $\sigma^{-1}(a) \in \mathbb{D}$ for all $a \in \mathbb{D}$. We write

$$\hat{\mathbf{F}}(\hat{Z}) := \mathbf{F}(\hat{Z}^{-1}) \cdot \hat{Z}^{N}, \tag{4.8}$$

and let $\hat{\mathbf{M}}_k(\hat{Z})$, $\hat{\mathbf{R}}_k(\hat{Z})$, $\vec{\mu}_k$, and $\vec{\omega}_k$ be the intermediate results obtained from the FFReduce algorithm with the input $\hat{\mathbf{F}}(\hat{Z})$. If we define

$$\mathbf{U}_k(Z) = Z^{\vec{\mu}_k} \cdot \hat{\mathbf{M}}_k(\hat{Z}), \qquad \mathbf{T}_k(Z) = Z^{\vec{\mu}_k} \cdot \hat{\mathbf{R}}_k(\hat{Z}) \cdot \hat{Z}^{\vec{\omega}_k - N \cdot \vec{e}}, \tag{4.9}$$

then $\mathbf{U}_k(Z) \cdot \mathbf{F}(Z) = \mathbf{T}_k(Z)$. In this case simple algebra shows that the recursion formulas for $\mathbf{U}_k(Z)$ obtained from (3.16) and (3.17) become

$$\sigma^{\vec{\mu}_k^{(\ell)}}(p_{\pi_k}) \cdot \mathbf{U}_{k+1}(Z)^{(\ell,\cdot)} = \sigma^{\vec{\mu}_k^{(\ell)}}(r_{\pi_k}) \cdot \mathbf{U}_k(Z)^{(\ell,\cdot)} - \sigma^{\vec{\mu}_k^{(\ell)}}(r_\ell) \cdot Z^{\vec{\mu}_k^{(\ell)} - \vec{\mu}_k^{(\pi_k)}} \cdot \mathbf{U}_k(Z)^{(\pi_k,\cdot)} \tag{4.10}$$

for $\ell \neq \pi_k$ and

$$\begin{aligned} &\sigma^{\vec{\mu}_k^{(\pi_k)}+2}(p_{\pi_k}) \cdot \mathbf{U}_{k+1}(Z)^{(\pi_k,\cdot)} \\ &= \sigma^{\vec{\mu}_k^{(\pi_k)}+1}(r_{\pi_k}) \cdot \mathbf{U}_k(Z)^{(\pi_k,\cdot)} - \sum_{\ell \neq \pi_k} \sigma^{\vec{\mu}_k^{(\pi_k)}+2}(p_\ell) \cdot Z^{\vec{\mu}_k^{(\pi_k)} - \vec{\mu}_k^{(\ell)} + 1} \cdot \mathbf{U}_{k+1}(Z)^{(\ell,\cdot)}, \end{aligned} \tag{4.11}$$

where

$$r_\ell = \sigma^{-\vec{\mu}_k^{(\ell)}}\left(c_{N+\vec{\mu}_k^{(\ell)}-\lfloor k/s \rfloor}\left(\mathbf{T}_k(Z)^{(\ell,(k \bmod m)+1)}\right)\right),$$
$$p_\ell = \sigma^{-\vec{\mu}_k^{(\pi_k)}}\left(c_{\vec{\mu}_k^{(\pi_k)}-\vec{\mu}_k^{(\ell)}-\delta_{\pi_k,\ell}+1}\left(\mathbf{U}_k(Z)^{(\pi_k,\ell)}\right)\right).$$

Since $\vec{\mu}_k^{(\pi_k)} \leq \vec{\mu}_k^{(\ell)}$ whenever $r_\ell \neq 0$ and $p_\ell = 0$ whenever $\vec{\mu}_k^{(\pi_k)} < \vec{\mu}_k^{(\ell)} - 1$ by the definition of a reduced order basis, it follows that $\mathbf{U}_{k+1}(Z) \in \mathbb{D}[Z;\sigma]^{m\times m}$ and hence $\mathbf{T}_{k+1}(Z) \in \mathbb{D}[Z;\sigma]^{m\times s}$. Moreover, $[\mathbf{U}_{k+1}(Z),\ \mathbf{T}_{k+1}(Z)]$ is obtained from $[\mathbf{U}_k(Z),\ \mathbf{T}_k(Z)]$ by elementary row operations of the second type, so if $\mathbf{U}_k(Z)$ is unimodular then $\mathbf{U}_{k+1}(Z)$ is also unimodular.

Theorem 4.10 *Let $k = \kappa s$, and $\hat{\mathbf{M}}_k(\hat{Z})$, $\hat{\mathbf{R}}_k(\hat{Z})$, $\vec{\mu}_k$, and $\vec{\omega}_k = \kappa \cdot \vec{e}$ be the final output obtained from the* FFREDUCE *algorithm with the input $\hat{\mathbf{F}}(\hat{Z})$. Then*

(a) $\mathbf{U}_k(Z) \in \mathbb{D}[Z;\sigma]^{m\times m}$ and $\mathbf{T}_k(Z) \in \mathbb{D}[Z;\sigma]^{m\times s}$;

(b) $\mathbf{U}_k(Z)$ is unimodular;

(c) $\mathbf{U}_k(Z) \cdot \mathbf{F}(Z) = \mathbf{T}_k(Z)$;

(d) the nonzero rows of $\mathbf{T}_k(Z)$ form a row-reduced matrix.

Proof. Parts (a), (b), and (c) have already been shown above. By Theorem 4.4, we see that rank $\hat{\mathbf{R}}_k(0) =$ rank $\hat{\mathbf{F}}(\hat{Z}) =$ rank $\hat{\mathbf{R}}_k(\hat{Z})$, which is also the number of nonzero rows in $\hat{\mathbf{R}}_k(\hat{Z})$. Therefore, the nonzero rows of $\hat{\mathbf{R}}_k(\hat{Z})$ form a matrix with trailing coefficient of full row rank. It is easy to see that rdeg $\mathbf{T}_k(Z) = \vec{\mu}_k + (N-\kappa)\cdot\vec{e}$

and that

$$\mathbf{T}_k(Z)^{(i,\cdot)} = \sigma^{\vec{\mu}_k^{(i)}}(\hat{\mathbf{R}}_k(0)^{(i,\cdot)}) \cdot Z^{\vec{\mu}_k^{(i)}+N-\kappa} + \text{lower degree terms.}$$

Therefore, $\mathrm{LC}_{row}(\mathbf{T}_k(Z)) = \sigma^{\deg \mathbf{T}_k(Z)-N+\kappa}(\hat{\mathbf{R}}_k(0))$. Since σ is an automorphism on $\mathbb{Q}_{\mathbb{D}}$, it follows that rank $\mathrm{LC}_{row}(\mathbf{T}_k(Z)) =$ rank $\hat{\mathbf{R}}_k(0)$, and hence the nonzero rows of $\mathbf{T}_k(Z)$ form a row-reduced matrix. □

We remark that Theorem 3.5 implies that the rows of $\mathbf{U}_k(Z)$ of Theorem 4.10 corresponding to the zero rows of $\mathbf{T}_k(Z)$ gives a basis of the left nullspace of $\mathbf{F}(Z)$.

4.4 Computing a Weak Popov Form

The FFReduce algorithm can be modified to obtain $\mathbf{U}(Z)$ and $\mathbf{T}(Z)$ such that $\mathbf{T}(Z)$ is in weak Popov form (Definition 2.7). Formally, if $\vec{\omega} = \kappa \cdot \vec{e}$ is the order obtained at the end of the FFReduce algorithm, we form the matrices $\mathbf{U}(Z)$ and $\mathbf{T}(Z)$ by

$$\begin{aligned}&[\mathbf{U}(Z)^{(i,j)},\ \mathbf{T}(Z)^{(i,j)}]\\ &= \begin{cases}[\mathbf{U}_k(Z)^{(i,j)},\ \mathbf{T}_k(Z)^{(i,j)}] & \text{if } \pi_k = i \text{ for some } \kappa s - s \le k < \kappa s,\\ [\mathbf{U}_{\kappa s}(Z)^{(i,j)},\ \mathbf{T}_{\kappa s}(Z)^{(i,j)}] & \text{otherwise;}\end{cases}\end{aligned}$$

We note that $\mathbf{U}(Z)$ and $\mathbf{T}(Z)$ are well-defined because the pivots π_k are distinct for $\kappa s - s \le k < \kappa s$ by Lemma 4.3. We now show that $\mathbf{T}(Z)$ is in weak Popov form.

Theorem 4.11 *Let $\vec{\omega} = \kappa \cdot \vec{e}$ be the order obtained from the* FFReduce *algorithm with the input $\hat{\mathbf{F}}(\hat{Z})$. Then*

(a) $\mathbf{U}(Z) \in \mathbb{D}[Z;\sigma]^{m\times m}$ *and* $\mathbf{T}(Z) \in \mathbb{D}[Z;\sigma]^{m\times s}$*;*

(b) $\mathbf{U}(Z)$ *is unimodular;*

(c) $\mathbf{U}(Z) \cdot \mathbf{F}(Z) = \mathbf{T}(Z)$*;*

(d) $\mathbf{T}(Z)$ *is in weak Popov form.*

Proof. Part (a) is clear, and (b) follows from the fact that $\mathbf{U}(Z)$ can be obtained from $\mathbf{U}_{\kappa s-s}(Z)$ by applying elementary row operations of the second type on each row until it has been chosen as a pivot. Moreover, we have that for all k and ℓ, $\mathbf{U}_k(Z)^{(\ell,\cdot)} \cdot \mathbf{F}(Z) = \mathbf{T}_k(Z)^{(\ell,\cdot)}$ and hence (c) is true.

Note that the non-pivot rows of $\mathbf{T}(Z)$ must be zero because of the termination condition (4.1). By reversing the coefficients of $\mathbf{T}(Z)$ we see that

$$\mathbf{T}(Z)^{(i,\cdot)} = \sigma^{\vec{\mu}^{(i)}_{\kappa s-s}}(H^{(i,\cdot)}) \cdot Z^{\vec{\mu}^{(i)}_{\kappa s-s}+N-\kappa} + \text{lower degree terms},$$

where H is the matrix whose nonzero rows are the pivot rows used during the last cycle.

By Lemma 4.2, the nonzero rows of H form a matrix in upper echelon form (up to row permutations). Thus, $\mathrm{LC}_{row}\left(\mathbf{T}(Z)\right) = \sigma^{\deg \mathbf{T}(Z)-N+\kappa}(H)$. Since σ is an automorphism on $\mathbb{Q}_{\mathbb{D}}$ it follows that the nonzero rows of $\mathrm{LC}_{row}\left(\mathbf{T}(Z)\right)$ is also in upper echelon form. Thus, $\mathbf{T}(Z)$ is in weak Popov form (see also Remark 2.10). □

Again, Theorem 3.5 implies that the rows of $\mathbf{U}(Z)$ of Theorem 4.11 corresponding to the zero rows of $\mathbf{T}(Z)$ gives a basis of the left nullspace of $\mathbf{F}(Z)$.

4.5 Computing GCRD and LCLM

Using the preceding algorithm for row reduction allows us to compute a greatest common right divisor (GCRD) and a least common left multiple (LCLM) of matrices of shift polynomials in the same way it is done in the case of matrices of polynomials [12, 41]. Let $\mathbf{A}(Z) \in \mathbb{D}[Z;\sigma]^{m_1 \times s}$ and $\mathbf{B}(Z) \in \mathbb{D}[Z;\sigma]^{m_2 \times s}$, such that the matrix

$$\mathbf{F}(Z) = \begin{bmatrix} \mathbf{A}(Z) \\ \mathbf{B}(Z) \end{bmatrix}$$

has rank s. Such an assumption is natural since otherwise we may have GCRDs of arbitrarily high degree [41, page 376]. After row reduction and possibly a permutation of the rows, we obtain

$$\mathbf{U}(Z) \cdot \mathbf{F}(Z) = \begin{bmatrix} \mathbf{U}_{11}(Z) & \mathbf{U}_{12}(Z) \\ \mathbf{U}_{21}(Z) & \mathbf{U}_{22}(Z) \end{bmatrix} \cdot \begin{bmatrix} \mathbf{A}(Z) \\ \mathbf{B}(Z) \end{bmatrix} = \begin{bmatrix} \mathbf{G}(Z) \\ 0 \end{bmatrix}$$

with $\mathbf{G}(Z) \in \mathbb{D}[Z;\sigma]^{s \times s}$, and $\mathbf{U}_{1,j}(Z)$, $\mathbf{U}_{2,j}(Z)$ matrices of shift polynomials of size $s \times m_j$, and $(m_1 + m_2 - s) \times m_j$, respectively, for $j = 1, 2$.

To see that $\mathbf{G}(Z)$ is a right divisor of $\mathbf{A}(Z)$ and $\mathbf{B}(Z)$, we use the fact that $\mathbf{U}(Z)$ is unimodular, so that an inverse $\mathbf{V}(Z)$ exists. Then

$$\mathbf{F}(Z) = \begin{bmatrix} \mathbf{A}(Z) \\ \mathbf{B}(Z) \end{bmatrix} = \mathbf{V}(Z) \cdot \begin{bmatrix} \mathbf{G}(Z) \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{V}_{11}(Z) & \mathbf{V}_{12}(Z) \\ \mathbf{V}_{21}(Z) & \mathbf{V}_{22}(Z) \end{bmatrix} \cdot \begin{bmatrix} \mathbf{G}(Z) \\ 0 \end{bmatrix},$$

so that $\mathbf{A}(Z) = \mathbf{V}_{11}(Z) \cdot \mathbf{G}(Z)$ and $\mathbf{B}(Z) = \mathbf{V}_{21}(Z) \cdot \mathbf{G}(Z)$. From

$$\mathbf{U}_{11}(Z) \cdot \mathbf{A}(Z) + \mathbf{U}_{12}(Z) \cdot \mathbf{B}(Z) = \mathbf{G}(Z)$$

we see that any right divisor of $\mathbf{A}(Z)$ and $\mathbf{B}(Z)$ must be a right divisor of $\mathbf{G}(Z)$. This shows that $\mathbf{G}(Z)$ is a GCRD of $\mathbf{A}(Z)$ and $\mathbf{B}(Z)$. Since $\mathbf{U}(Z)$ is unimodular, it follows that $\mathbf{U}_{21}(Z)$ and $\mathbf{U}_{22}(Z)$ are left coprime. In the case of polynomial matrices with $\mathbf{A}(Z)$ being nonsingular, the matrix fraction $\mathbf{B}(Z) \cdot \mathbf{A}(Z)^{-1}$ can be represented by the irreducible matrix fraction $\mathbf{U}_{22}^{-1}(Z) \cdot \mathbf{U}_{21}(Z)$.

Let $\ell > 0$. Then for any common left multiple $\mathbf{W}_1(Z) \cdot \mathbf{A}(Z) = \mathbf{W}_2(Z) \cdot \mathbf{B}(Z)$ with $\mathbf{W}_j(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma]^{\ell \times m_j}$, the rows of $[\mathbf{W}_1(Z),\ -\mathbf{W}_2(Z)]$ belong to the left nullspace $\mathcal{N}_{\mathbf{F}(Z)}$. Since $[\mathbf{U}_{21}(Z),\ \mathbf{U}_{22}(Z)]$ is a basis of $\mathcal{N}_{\mathbf{F}(Z)}$ by Theorem 3.5, there exists $\mathbf{Q}(Z) \in \mathbb{Q}_{\mathbb{D}}[Z;\sigma]^{\ell \times (m_1+m_2-s)}$ such that

$$\begin{bmatrix} \mathbf{W}_1(Z) & -\mathbf{W}_2(Z) \end{bmatrix} = \mathbf{Q}(Z) \cdot \begin{bmatrix} \mathbf{U}_{21}(Z) & \mathbf{U}_{22}(Z) \end{bmatrix},$$

implying that $\mathbf{U}_{21}(Z) \cdot \mathbf{A}(Z) = -\mathbf{U}_{22}(Z) \cdot \mathbf{B}(Z)$ is an LCLM of $\mathbf{A}(Z)$ and $\mathbf{B}(Z)$.

We remark that in contrast to the method proposed in [12], our GCRD has the additional property of being row-reduced or being in weak Popov form.

4.6 Computing Subresultants

The method of Section 4.5, applied to two 1×1 matrices, gives the GCRD and LCLM of two shift polynomials $a(Z)$ and $b(Z)$. In this section we examine the relationship of the intermediate results obtained during our algorithm to the subresultants of shift polynomials defined by [44, 45]. Denoting the degrees of $a(Z)$, $b(Z)$ by $d_a \geq d_b \geq 1$, the jth subresultant $\mathrm{sres}_j(a,b)$ for shift polynomials is defined by taking the determinant of the matrix

$$\begin{bmatrix}
\sigma^{d_b-j-1}(a_{d_a}) & \sigma^{d_b-j-1}(a_{d_a-1}) & \cdots & \cdots & \cdots & \sigma^{d_b-j-1}(a_{2j+2-d_b}) & Z^{d_b-j-1}\cdot a(Z) \\
 & \ddots & & & & \vdots & \vdots \\
 & & \sigma(a_{d_a}) & \cdots & \cdots & \sigma(a_j) & Z\cdot a(Z) \\
 & & & a_{d_a} & \cdots & a_{j+1} & a(Z) \\
\sigma^{d_a-j-1}(b_{d_b}) & \sigma^{d_a-j-1}(b_{d_b-1}) & \cdots & \cdots & \cdots & \sigma^{d_a-j-1}(b_{2j+2-d_a}) & Z^{d_a-j-1}\cdot b(Z) \\
 & \ddots & & & & \vdots & \vdots \\
 & & \sigma(b_{d_b}) & \cdots & \cdots & \sigma(b_j) & Z\cdot b(Z) \\
 & & & b_{d_b} & \cdots & b_{j+1} & b(Z)
\end{bmatrix}.$$

Theorem 4.12 *Let $a(Z)$ and $b(Z)$ be two shift polynomials of degrees d_a and d_b, respectively, such that $d_a \geq d_b \geq 1$. Then $\mathit{sres}_j(a,b) \neq 0$ if and only if there exists an $\ell = \ell_j$ with $\vec{\mu}_{2d_a-2j-1} = (d_a - j, d_a - j) - \vec{e}_\ell$. In this case,*

$$\mathbf{T}_{2d_a-2j-1}(Z)^{(\ell,1)} = \pm\gamma \cdot \mathit{sres}_j(a,b), \qquad \gamma = \prod_{i=0}^{d_a-d_b-1} \sigma^{d_b-j+i}(a_{d_a}).$$

Proof. After expanding with respect to the first columns, we see that the quantity $\gamma \cdot \mathrm{sres}_j(a,b)$ coincides with the determinant of the matrix

$$\begin{bmatrix}
\sigma^{d_a-j-1}(a_{d_a}) & \sigma^{d_a-j-1}(a_{d_a-1}) & \cdots & \cdots & \cdots & \sigma^{d_a-j-1}(a_{2j+2-d_a}) & Z^{d_a-j-1}\cdot a(Z) \\
 & \ddots & & & & \vdots & \vdots \\
 & & \sigma(a_{d_a}) & \cdots & \cdots & \sigma(a_j) & Z\cdot a(Z) \\
 & & & a_{d_a} & \cdots & a_{j+1} & a(Z) \\
\sigma^{d_a-j-1}(b_{d_a}) & \sigma^{d_a-j-1}(b_{d_a-1}) & \cdots & \cdots & \cdots & \sigma^{d_a-j-1}(b_{2j+2-d_a}) & Z^{d_a-j-1}\cdot b(Z) \\
 & \ddots & & & & \vdots & \vdots \\
 & & \sigma(b_{d_a}) & \cdots & \cdots & \sigma(b_j) & Z\cdot b(Z) \\
 & & & b_{d_b} & \cdots & b_{j+1} & b(Z)
\end{bmatrix}.$$

Denote by S_j the $(2d_a-2j)\times(2d_a-2j-1)$ matrix obtained by dropping the last column. Notice that

$$\sigma^{-(d_a-j-1)}(S_j) = K((d_a-j,d_a-j),2d_a-2j-1), \tag{4.12}$$

the striped Krylov matrix associated to $\hat{\mathbf{F}}(\hat{Z}) = [\hat{a}(\hat{Z}),\ \hat{b}(\hat{Z})]^T$ with $\hat{a}(\hat{Z}) = a(\hat{Z}^{-1})\cdot \hat{Z}^{d_a}$ and $\hat{b}(\hat{Z}) = b(\hat{Z}^{-1})\cdot\hat{Z}^{d_a}$. Thus $\mathrm{sres}_j(a,b)\neq 0$ if and only if the dimension (over $\mathbb{Q}_{\mathbb{D}}$) of the left nullspace of S_j is equal to one, which in turn is true if and only if there is a unique $\mathbf{P}(Z)\in\mathbb{Q}_{\mathbb{D}}[Z;\sigma]$ (up to multiplication with an element from $\mathbb{Q}_{\mathbb{D}}$) of order $\vec{\omega} = (2d_a-2j-1)$ and $\deg \mathbf{P}(Z)\le d_a-j-1$.

One verifies using Lemma 5.2 of [10] and $d_a\neq 0$ that $|\vec{\omega}_k| = k = |\vec{\mu}_k|$ for all k in the FFReduce algorithm. Let $k = 2d_a-2j-1$, then from Lemma 3.3(a) and Definition 3.10(b) we conclude that $\mathrm{sres}_j(a,b)\neq 0$ if and only if $\vec{\mu}_k$ has one component being equal to d_a-j-1 and the other one being at least as large as

$d_a - j$, that is, $\vec{\mu}_k = (d_a - j, d_a - j) - \vec{e}_\ell$ for some $\ell \in \{1, 2\}$.

Finally, if $\mathrm{sres}_j(a, b) \neq 0$, then we use (4.12) and the determinantal representations of Section 3.3 together with the uniqueness of Mahler systems in order to conclude that

$$\gamma \cdot \mathrm{sres}_j(a, b) = \pm Z^{\vec{\mu}^{(\ell)}} \cdot \hat{\mathbf{R}}_k(\hat{Z})^{(\ell,\cdot)} \cdot \hat{Z}^{\vec{\omega} - d_a \cdot \vec{e}} = \mathbf{T}_k(Z)^{(\ell,1)}.$$

$\square$

Whenever $\vec{\mu}_{2k-1}$ is of the form $(k, k) - \vec{e}_\ell$ for some $\ell \in \{1, 2\}$ during the execution of our algorithm, we can recover the nonzero $\mathrm{sres}_{d_a - k}(a, b)$ from $\hat{\mathbf{R}}_{2k-1}(\hat{Z}) \cdot Z^{\vec{\omega} - d_a \cdot \vec{e}}$ after multiplying by Z^k and dividing by the extra factor of γ (or by dividing $\mathbf{T}_{2k-1}(Z)^{(\ell,1)}$ by γ).

Notice that the extra factor of γ is introduced at the beginning of the algorithm, before any step with $|\Lambda| > 1$. There is no reduction performed in these first $d_a - d_b$ steps. Thus, we may modify our algorithm so that no reduction is done until $|\Lambda| = 2$ for the first time, except that $\vec{\mu}_k$ is still updated. Then

$$\mathrm{sres}_{d_a - k}(a, b) = \begin{cases} \pm Z^{\vec{\mu}^{(1)}_{2k-1} - d_a + d_b} \cdot \hat{\mathbf{R}}_{2k-1}(\hat{Z})^{(1,1)} \cdot \hat{Z}^{2k-1-d_a} & \text{if } \vec{\mu}_{2k-1} = (k-1, k), \\ \pm Z^{\vec{\mu}^{(2)}_{2k-1}} \cdot \hat{\mathbf{R}}_{2k-1}(\hat{Z})^{(2,1)} \cdot \hat{Z}^{2k-1-d_a} & \text{if } \vec{\mu}_{2k-1} = (k, k-1). \end{cases}$$

Chapter 5

A Modular Algorithm for Row-Reduced Form for Polynomial Matrices

In this chapter, we give a modular algorithm for computing a row-reduced form of a polynomial matrix. The modular algorithm is based on the fraction-free FFReduce algorithm given in Algorithm 3.5.

For simplicity, we assume that $\mathbb{D} = \mathbb{Z}$ or $\mathbb{D} = \mathbb{Q}_R[x]$ for some integral domain R. Given an ideal $I \subseteq \mathbb{D}$, a modular homomorphism $\phi : \mathbb{D} \rightarrow \mathbb{D}/I$ can be defined by $\phi(a) = a + I$ for all $a \in \mathbb{D}$. For example, if $\mathbb{D} = \mathbb{Z}$ then we set $I = p\mathbb{Z}$ for some prime $p \in \mathbb{Z}$, so that ϕ reduces integers modulo p and $\mathbb{D}/I = \mathbb{Z}_p$. On the other hand, if $\mathbb{D} = \mathbb{Q}_R[x]$, then we set $I = (x - \alpha)\mathbb{Q}_R[x]$ for some $\alpha \in R$ (or $\alpha \in \mathbb{Q}_R$), so that $\mathbb{D}/I = \mathbb{Q}_R$ and the modular homomorphism corresponds to evaluation at $x = \alpha$. We will denote the reduction homomorphism by ϕ_p or $\phi_{x-\alpha}$ if we wish to explicitly specify the ideal I. Finally, if $\mathbb{D}$ is a multivariate polynomial ring over $\mathbb{Z}$, our modular algorithm can be applied recursively to eliminate one variable at a time and reduce the coefficients to $\mathbb{Z}_p$. We may also first reduce $\mathbb{D}$ to a multivariate polynomial ring over $\mathbb{Z}_p$ and then eliminate the variables.

5.1 Issues in Designing a Modular Algorithm

The basic framework of a modular algorithm can be stated as follows [36]. First, a number of pairwise comaximal ideals $I_1, \ldots, I_k \subseteq \mathbb{D}$ (i.e. $I_i + I_j = \mathbb{D}$ if $i \neq j$) [30] are chosen. In our case, pairwise comaximality is guaranteed by choosing distinct primes (when $\mathbb{D} = \mathbb{Z}$) or distinct evaluation points (when $\mathbb{D} = \mathbb{Q}_R[x]$). Let ϕ_i be the modular homomorphism defined by the ideal I_i. For $i = 1, \ldots, k$, a row-reduced form $\mathbf{T}_i(z)$ of $\phi_i(\mathbf{F}(z)) \in (\mathbb{D}/I_i)[z]^{m \times s}$ is computed. At the end, those $\mathbf{T}_i(z)$ which are images of the desired result $\mathbf{T}(z) \in \mathbb{D}[z]^{m \times s}$ are used to reconstruct $\mathbf{T}(z)$ by Chinese remaindering (e.g. Garner's algorithm [32]). The unimodular transformation matrix $\mathbf{U}(z)$ can also be reconstructed in a similar way from $\mathbf{U}_i(z)$.

In order to design a modular algorithm, we must recognize when the computed result $\mathbf{T}_i(z)$ is an image of the desired result. That is, we need to recognize whether $\mathbf{T}_i(z) = \phi_i(\mathbf{T}(z))$. This equality may not hold for two reasons. First, a polynomial matrix can have many different row-reduced forms, so that $\mathbf{T}_i(z)$ may be the homomorphic image of another row-reduced form $\mathbf{T}'(z)$ for $\mathbf{F}(z)$. This can be addressed by *normalization*, so that the computed row-reduced form satisfies $\mathbf{T}_i(z) = \phi_i(\mathbf{T}(z))$. Secondly, the computed result $\mathbf{T}_i(z)$ may not be the homomorphic image of *any* row-reduced form of $\mathbf{F}(z)$. If we cannot normalize the result so that $\mathbf{T}_i(z) = \phi_i(\mathbf{T}(z))$, we say that ϕ_i is an *unlucky homomorphism* and the computed result $\mathbf{T}_i(z)$ is discarded. For example, if $\mathbf{F}(z)$ is square and $\deg \phi_i(\det \mathbf{F}(z)) < \deg \det \mathbf{F}(z)$, then the row degree of a row-reduced form of $\mathbf{F}(z)$ in $\mathbb{D}/I_i$ will be different from rdeg $\mathbf{T}(z)$ (under any row permutation). In such cases, $\mathbf{T}_i(z)$ does not correspond to any row-reduced form of $\mathbf{F}(z)$. Thus, we must be able to recognize when ϕ_i is unlucky and discard the corresponding results.

Since the Chinese remainder theorem is only able to guarantee correctness of the reconstructed results in $\mathbb{D}/(I_1 \cup \cdots \cup I_k) = \mathbb{D}/(I_1 \cdots I_k)$ [30], additional information is required to ensure that the reconstructed results are correct in $\mathbb{D}$. Typically, bounds on the size of the results are obtained and the number of lucky homomorphic images is determined. For a large enough k, there is only one representative in the coset in $\mathbb{D}/(I_1 \cdots I_k)$ satisfying the bounds. This representative gives the correct answer. Thus, we must also determine the number of lucky homomorphic images required.

We address each of these issues with linear algebra techniques, since the FFREDUCE algorithm can be viewed as an algorithm for solving specific systems of linear equations.

5.2 Computing Homomorphic Images

We view the domain $\mathbb{D}/I$ as a field in which coefficient growth is limited. Therefore, a number of different algorithms can be used to compute a row-reduced form for $\mathbf{F}(z)$ over $\mathbb{D}/I$. In our algorithm we choose the FFREDUCE algorithm given in Algorithm 3.5 with the modified termination condition given in (4.1). We also need to reverse the coefficients using (4.8) and (4.9), or rewrite the algorithm using the recurrence formulas (3.16) and (3.17). To simplify our presentation, we will assume that the coefficients are reversed in the algorithm before any computation is done, and the output is reversed at the end. However, to simplify our analysis, we will study the reversed transformation matrix $\mathbf{M}(z)$ and residual $\mathbf{R}(z)$ instead of $\mathbf{U}(z)$ and $\mathbf{T}(z)$. Note that the reversal of coefficients is trivial in the case of polynomial matrices, and does not involve operations on the coefficients.

Since the multi-gradient d of a Mahler system is defined only up to sign, the computed images may have incorrect signs. Therefore, we insist that $d = \det K^*(\vec{\mu}-\vec{e}, \vec{\omega})$ to ensure that the sign is correct. This is done in the FFReduce algorithm by keeping track of the sign ϵ_k. We have $\epsilon_0 = 1$, and the update formula [12]

$$\epsilon_{k+1} \leftarrow \begin{cases} \epsilon_k & \text{if } \Lambda = \{\}, \\ \epsilon_k \cdot (-1)^{\sum_{i=\pi+1}^{n} \vec{\mu}_k^{(i)}} & \text{otherwise.} \end{cases}$$

We also record in a vector $\vec{\sigma}_k$ the values of $i+1$ such that $\Lambda \neq \{\}$ in iteration i. Thus, $\vec{\sigma}_k$ has $|\vec{\mu}_k|$ components, and $\vec{\sigma}_k^{(j)} = i+1$ if $\Lambda \neq \{\}$ in iteration i and $j = |\vec{\mu}_i| + 1$. Then, the final results returned are modified to be

$$\mathbf{M} \leftarrow \epsilon_k \cdot \mathbf{M}_k, \quad \mathbf{R} \leftarrow \epsilon_k \cdot \mathbf{R}_k, \quad \vec{\sigma} \leftarrow \vec{\sigma}_k.$$

We remark that $\vec{\sigma}$ gives the elements of the set of column indices J in Definition 2.16.

We supply the additional input parameter $\mathbb{D}/I$ to the FFReduce algorithm to specify the domain of computation. We perform exactly the same arithmetic operations as stated in Algorithm 3.5 in $\mathbb{D}/I$, where the division of the predicted divisor is replaced by multiplication of the inverse in the field. It is unusual to use a fraction-free algorithm for computation over $\mathbb{D}/I$ because coefficient growth is limited. However, this ensures that the computed results are images of the results computed in $\mathbb{D}$ using the same sequence of operations. This solves part of our normalization problem. In the remainder of this chapter, we call the modified algorithm FFReduce2.

5.3 Lucky Homomorphisms and Normalization

We define lucky homomorphisms in this section. Let $\mathbf{M}(z)$, $\mathbf{R}(z)$, $\vec{\mu}$, $\vec{\omega}$, and $\vec{\sigma}$ be the results obtained by the FFReduce2 algorithm when the operations are performed over $\mathbb{D}$. Similarly, let $\mathbf{M}_i(z)$, $\mathbf{R}_i(z)$, $\vec{\mu}_i$, $\vec{\omega}_i$, and $\vec{\sigma}_i$ be the results computed over $\mathbb{D}/I_i$[1]. We also define d and d_i to be the normalization constant in Definition 3.10(c) corresponding to $\mathbf{M}(z)$ and $\mathbf{M}_i(z)$, respectively. Note that $\mathbf{M}_i(z)$ and $\mathbf{R}_i(z)$ can be used in the reconstruction if

$$\phi_i(\mathbf{M}(z)) = \mathbf{M}_i(z), \quad \phi_i(\mathbf{R}(z)) = \mathbf{R}_i(z). \tag{5.1}$$

If this is not the case, we say that ϕ_i is unlucky. Since we require the computed results to be the exact images of $\mathbf{M}(z)$ and $\mathbf{R}(z)$, we have solved the normalization problem as well provided that ϕ_i is lucky. Formally, we define a lucky homomor-

[1]These quantities are not the intermediate results computed in the FFReduce2 algorithm, although the notation is the same.

phism in the following way.

Definition 5.1 *Let ϕ_i be a modular homomorphism. Then ϕ_i is* lucky *if $\phi_i(d) \neq 0$ and $|\vec{\mu}| = |\vec{\mu}_i|$. Otherwise, ϕ_i is* unlucky. □

We also say that a prime p or an evaluation point α is lucky (or unlucky) if the corresponding modular homomorphism is lucky (or unlucky). We remark that if $\deg \phi_i(\mathbf{F}(z)) < \deg \mathbf{F}(z)$, then ϕ_i is unlucky because column $\vec{\sigma}^{(1)}$ of $\phi_i(K^*(\vec{\mu}-\vec{e},\vec{\omega}))$ consists only of zeros and therefore $\phi_i(d) = 0$ (since the coefficients are reversed).

Before we prove that this definition is sufficient (i.e. (5.1) is satisfied) and show how to detect whether a modular homomorphism is unlucky, we need to state an additional property satisfied by the degrees of the Mahler systems computed in the FFReduce2 algorithm [12, Theorem 7.3]. Roughly speaking, this describes how the sequence of row indices of pivot rows deviates from the "normal" sequence in which all residuals are nonzero at every iteration.

Theorem 5.2 *Let $w = \{\vec{w}_k\}_{k=0,1,2\ldots}$ be the sequence of multi-indices defined by*

$$\begin{aligned} \vec{w}_0 &= \vec{0} \\ \vec{w}_{k+1} &= \vec{w}_k + \vec{e}_\pi, \text{ where } \pi = \min_{1\le i\le m}\left\{ i : \vec{w}_k^{(i)} = \min_{1\le j\le m} \vec{w}_k^{(j)} \right\}. \end{aligned}$$

Then $\vec{\mu}$ is the unique closest point to the sequence w such that $K^*(\vec{\mu}-\vec{e},\vec{\omega})$ is nonsingular. That is, if $K^*(\vec{\nu}-\vec{e},\vec{\omega})$ is nonsingular for some $\vec{\nu}$ such that $|\vec{\nu}| = |\vec{\mu}|$, then

$$\left|\max\{\vec{0}, \vec{w}_k - \vec{\mu}\}\right| \le \left|\max\{\vec{0}, \vec{w}_k - \vec{\nu}\}\right| \qquad \text{for } k \ge 0. \tag{5.2}$$

□

To facilitate the presentation, we let $K_i^*(\vec{\mu}-\vec{e},\vec{\omega},\phi_i(\mathbf{F}(z)))$ be the submatrix of the corresponding striped Krylov matrix $K_i(\vec{\mu}-\vec{e},\vec{\omega},\phi_i(\mathbf{F}(z)))$ over $\mathbb{D}/I$ as defined in Definition 2.16, so that the set of column indices J is given by $\vec{\sigma}_i$. In the following, we will often make use of the following fact.

Lemma 5.3 *If $|\vec{\mu}_i| = |\vec{\mu}|$, then the columns indexed by $\vec{\sigma}$ are also contained in $K_i(\vec{\mu}_i,\vec{\omega}_i,\phi_i(\mathbf{F}(z)))$. In other words, if $\vec{\omega}_i = \kappa \cdot \vec{e}$, then $\vec{\sigma}^{(k)} \leq \kappa$ for $k = 1, \ldots, |\vec{\mu}|$.*

Proof. If $\vec{\sigma}^{(k)} > \kappa$, then rank $K_i(\vec{\mu}_i,\vec{\omega}_i,\phi_i(\mathbf{F}(z))) < |\vec{\sigma}| = |\vec{\sigma}_i|$ by Definition 2.16, which is a contradiction. □

We now prove a lemma which will be used for detecting whether a homomorphism is lucky.

Lemma 5.4 *Suppose $\deg \mathbf{F}(z) = \deg \phi_i(\mathbf{F}(z))$ and $|\vec{\mu}_i| = |\vec{\mu}|$. Then $\vec{\sigma} \leq_{lex} \vec{\sigma}_i$. Moreover, if $\vec{\sigma} = \vec{\sigma}_i$, then $\vec{\mu}$ is at least as close to w as $\vec{\mu}_i$, as defined in (5.2).*

Proof. The columns indexed by $\vec{\sigma}_i$ in $K_i(\vec{\mu}_i-\vec{e},\vec{\omega}_i,\phi_i(\mathbf{F}(z)))$ are linearly independent over $\mathbb{D}/I$. Therefore, the same columns in $K(\vec{\mu}_i-\vec{e},\vec{\omega},\mathbf{F}(z))$ are also linearly independent over $\mathbb{Q}_{\mathbb{D}}$ by Lemma 5.3. By Definition 2.16, it follows that $\vec{\sigma} \leq_{lex} \vec{\sigma}_i$.

If $\vec{\sigma} = \vec{\sigma}_i$, then $\phi_i\left(\det K^*(\vec{\mu}_i-\vec{e},\vec{\omega},\mathbf{F}(z))\right) = \det K_i^*(\vec{\mu}_i-\vec{e},\vec{\omega}_i,\phi_i(\mathbf{F}(z))) = d_i \neq 0$, it follows that $K^*(\vec{\mu}_i-\vec{e},\vec{\omega},\mathbf{F}(z))$ is nonsingular over $\mathbb{Q}_{\mathbb{D}}$. The second part now follows from Theorem 5.2. □

We now give an equivalent definition of lucky homomorphisms which is more useful for the detection of unlucky homomorphisms.

Theorem 5.5 *Suppose $\deg \mathbf{F}(z) = \deg \phi_i(\mathbf{F}(z))$. Then ϕ_i is lucky if and only if $\vec{\mu}_i = \vec{\mu}$ and $\vec{\sigma}_i = \vec{\sigma}$.*

Proof. Suppose ϕ_i is lucky. Since $\phi_i(d) \neq 0$, $\phi_i(K^*(\vec{\mu}-\vec{e}, \vec{\omega}, \mathbf{F}(z)))$ is nonsingular over $\mathbb{D}/I$. Thus, the columns of $K(\vec{\mu}-\vec{e}, \vec{\omega}, \phi_i(\mathbf{F}(z)))$ indexed by $\vec{\sigma}$ are linearly independent over $\mathbb{D}/I$, so that $\vec{\sigma}_i \leq_{lex} \vec{\sigma}$. But $\vec{\sigma} \leq_{lex} \vec{\sigma}_i$ by Lemma 5.4, hence $\vec{\sigma}_i = \vec{\sigma}$. Moreover, $\vec{\mu}$ is at least as close to w as $\vec{\mu}_i$ by Lemma 5.4. On the other hand, $\vec{\mu}_i$ is the unique closest point to w by Theorem 5.2. This implies that $\vec{\mu}_i = \vec{\mu}$.

Conversely, assume that $\vec{\mu}_i = \vec{\mu}$ and $\vec{\sigma}_i = \vec{\sigma}$. Clearly $|\vec{\mu}| = |\vec{\mu}_i|$. In addition, $\phi_i(K^*(\vec{\mu}-\vec{e}, \vec{\omega}, \mathbf{F}(z))) = K_i^*(\vec{\mu}_i - \vec{e}, \vec{\omega}_i, \phi_i(\mathbf{F}(z)))$ by Lemma 5.3, so that $\phi_i(d) = d_i \neq 0$. □

We now show that Definition 5.1 is sufficient.

Theorem 5.6 *If ϕ_i is lucky, then $\phi_i(\mathbf{M}(z)) = \mathbf{M}_i(z)$ and $\phi_i(\mathbf{R}(z)) = \mathbf{R}_i(z)$.*

Proof. Suppose that ϕ_i is lucky, so that $\vec{\mu} = \vec{\mu}_i$ and $\vec{\sigma} = \vec{\sigma}_i$ by Theorem 5.5. Then $\phi_i(K^*(\vec{\mu}-\vec{e}+\vec{e}_j, \vec{\omega}, \mathbf{F}(z))) = K_i^*(\vec{\mu}_i - \vec{e}+\vec{e}_j, \vec{\omega}_i, \phi_i(\mathbf{F}(z)))$ for all j. It follows by (3.12) that $\phi_i(\mathbf{M}(z)) = \mathbf{M}_i(z)$. Finally, over $\mathbb{D}/I_i$ we have

$$\phi_i(\mathbf{R}(z)) = \phi_i(\mathbf{M}(z) \cdot \mathbf{F}(z)) = \phi_i(\mathbf{M}(z)) \cdot \phi_i(\mathbf{F}(z)) = \mathbf{M}_i(z) \cdot \phi_i(\mathbf{F}(z)) = \mathbf{R}_i(z).$$

□

To determine if ϕ_i is lucky, We need to check that $\vec{\mu}_i = \vec{\mu}$ and $\vec{\sigma}_i = \vec{\sigma}$. However, $\vec{\mu}$ and $\vec{\sigma}$ are not known in advance. Instead, we can compare the results computed under two modular homomorphisms and detect which, if any, is unlucky.

Theorem 5.7 *Suppose ϕ_i and ϕ_j satisfy* $\deg \mathbf{F}(z) = \deg \phi_i(\mathbf{F}(z)) = \deg \phi_j(\mathbf{F}(z))$. *Then ϕ_i is unlucky if either of the following holds:*

(a) $|\vec{\mu}_i| = |\vec{\mu}_j|$ and $\vec{\sigma}_i >_{lex} \vec{\sigma}_j$.

(b) $|\vec{\mu}_i| = |\vec{\mu}_j|$, $\vec{\sigma}_i = \vec{\sigma}_j$, *and* $\vec{\mu}_j$ *is closer to* w *than* $\vec{\mu}_i$*;*

Furthermore, if $|\vec{\mu}_i| \neq |\vec{\mu}_j|$, *then at least one of* ϕ_i *and* ϕ_j *is unlucky.*

Proof. Conditions (a) and (b) follow from Lemma 5.4. If $|\vec{\mu}_i| \neq |\vec{\mu}_j|$, then they cannot be both equal to $|\vec{\mu}|$, so at least one of ϕ_i and ϕ_j must be unlucky. □

Remark 5.8 *Note that when* $|\vec{\mu}_i| \neq |\vec{\mu}_j|$, *we cannot determine whether one of* ϕ_i *or* ϕ_j *is lucky by the criteria above. Thus, we must discard both* ϕ_i *and* ϕ_j.

If the termination condition of FFReduce2 *is the original termination condition of* FFReduce *(i.e. perform* $(mN+1)s$ *iterations), then we must have* $|\vec{\mu}_i| \leq |\vec{\mu}|$ *in all cases. This is because* $\vec{\omega}_i = \vec{\omega}$ *for all* i, *and* $|\vec{\mu}|$ *is the maximum rank of* $K(\vec{\nu} - \vec{e}, \vec{\omega}, \mathbf{F}(z))$ *over all multi-indices* $\vec{\nu}$. *The rank of such matrices over* $\mathbb{D}/I$ *cannot increase, so that* $|\vec{\mu}_i| \leq |\vec{\mu}|$. *Using the original termination condition, we can discard only* ϕ_i *if* $|\vec{\mu}_i| < |\vec{\mu}_j|$. *Since unlucky homomorphisms are rarely encountered, the advantage of this approach is offset by the increased average running time of* FFReduce2. *Therefore, we will not consider this approach.* □

Finally, we need to bound the number of unlucky homomorphisms. If ϕ_i is unlucky, then either $\phi_i(d) = 0$ or $|\vec{\mu}| \neq |\vec{\mu}_i|$ by Definition 5.1. In the latter case, the required number of nonzero rows is obtained prematurely and the algorithm terminates, which implies that some of the determinants given by (3.13) vanishes under the homomorphism. In either case, we have $\phi_i(d') = 0$ where d'. Now d' is a minor of $K^*(\vec{\mu}, \vec{\omega})$ of order $\min(m,s)(mN+1)$. Each minor has size $\mathcal{O}(\min(m,s)(mN+1)K)$ by Lemma 3.22. If $\mathbb{D} = \mathbb{Z}$, the product of all unlucky primes must divide the product of the $m+1$ minors, which is in $\mathcal{O}(m\min(m,s)(mN+1)K)$. Similarly, the number of unlucky evaluation points is finite if $\mathbb{D} = \mathbb{Q}_R[x]$. In practice, unlucky homomorphisms are rarely encountered.

5.4 Number of Homomorphic Images Required

In order to determine the number of lucky homomorphisms required to guarantee the correctness of the reconstructed results, we need to obtain a bound on the size of the coefficients in $\mathbf{M}(z)$ and $\mathbf{R}(z)$. This was given in Lemma 3.22 and Theorem 4.6. We restate the result here for the special case of polynomial matrices.

Lemma 5.9 *Let K be a bound on the size of the coefficients appearing in $\mathbf{F}(z)$. Then the size of the coefficients in $\mathbf{M}(z)$ and $\mathbf{R}(z)$ is $\mathcal{O}(\min(m, s)mNK)$.*

Moreover, if $\mathbb{D} = \mathbb{Z}$ and K is a bound on the number of bits required to store each coefficient, then the magnitude of each coefficient of $\mathbf{M}(z)$ and $\mathbf{R}(z)$ is bounded by $n^{n/2}2^{nK}$ with $n = \min(m, s)(mN + 1)$. If $\mathbb{D} = \mathbb{Q}_R[x]$ and K is a bound on the degree of each coefficient, then the degree of each coefficient of $\mathbf{M}(z)$ and $\mathbf{R}(z)$ is bounded by $\min(m, s)(mN + 1)K$. □

Therefore, in the case of $\mathbb{D} = \mathbb{Z}$ we need to have enough lucky primes such that their product exceeds $2n^{n/2}2^{nK}$. Hence, the number of lucky primes required is $\mathcal{O}(\min(m, s)(mN + 1))$ where we assume that the primes chosen have size approximately K and $\log n < K$, which are usually satisfied. From the previous section, there are potentially $\mathcal{O}(m \min(m, s)(mN + 1))$ unlucky primes. As a result, we may have to use primes of size approximately mK to achieve the same number of primes. Alternatively, we may choose primes from a set whose size is at least three times the number of potential unlucky primes. Then the probability of encountering an unlucky prime is less than 1/2, and it is extremely unlikely to encounter $\Omega(\min(m, s)(mN + 1))$ unlucky primes. In the case of $\mathbb{D} = \mathbb{Q}_R[x]$ we need to have $\min(m, s)(mN + 1)K + 1$ lucky evaluation points.

5.5 Complete Algorithm and Complexity

We give the complete modular algorithm ModReduce in Algorithm 5.1. For simplicity, we give the algorithm only in the case of $\mathbb{D} = \mathbb{Z}$. The case $\mathbb{D} = \mathbb{Q}_R[x]$ is similar. We also assume that there is a CRA subroutine using Garner's algorithm that updates the reconstructed matrices by Chinese remaindering after each additional image has been computed.

Instead of terminating the algorithm after the product of primes exceeds $2n^{n/2}2^{nK}$ as implied by Lemma 5.9, we may instead terminate the algorithm when the reconstructed result does not change for one step, the nonzero rows of $\mathbf{R}(0)$ have full row rank, and the relation $\mathbf{U} \cdot \mathbf{F} = \mathbf{T}$ holds (after reversing coefficients). One way to guarantee that the nonzero rows of $\mathbf{R}(0)$ has full row rank is to ensure that $\mathbf{R}_i$ have zero rows at the same row indices. Although the computed results may not match the ones computed by the FFReduce algorithm over $\mathbb{Z}$, we can still obtain a unimodular transformation such that the result is row-reduced. This idea of early termination is similar to the trial division technique commonly used in the case of modular algorithms for polynomial GCD [36], and is useful in practice because the Hadamard's bound is usually too pessimistic.

Algorithm 5.1 The ModReduce Algorithm for Row-Reduced Form for $\mathbb{D} = \mathbb{Z}$

Input: Polynomial matrix $\mathbf{F} \in \mathbb{Z}[z]^{m \times s}$.

Output: Mahler system $\mathbf{M} \in \mathbb{Z}[z]^{m \times m}$ of degree $\vec{\mu}$ and order $\vec{\omega}$, and residual $\mathbf{R} \in \mathbb{Z}[z]^{m \times s}$.

```
Compute the bound of coefficient size $K$ from $\mathbf{F}$
$n \leftarrow \min(m, s)(mN + 1)$
$(i, q, \vec{\mu}, \vec{\sigma}, \mathbf{M}, \mathbf{R}) \leftarrow (1, 1, \vec{0}, \vec{0}, \mathbf{0}, \mathbf{0})$
while $q \le 2n^{n/2}2^{nK}$ do
  repeat
    $p \leftarrow$ a new prime of size $K$
  until $\deg \phi_p(\mathbf{F}) = \deg \mathbf{F}$
  $(\vec{\mu}_i, \vec{\sigma}_i, \mathbf{M}_i, \mathbf{R}_i) \leftarrow \mathrm{FFReduce2}(\phi_p(\mathbf{F}), \mathbb{Z}_p)$
  if $q = 1$ then
    $(q, \vec{\mu}, \vec{\sigma}, \mathbf{M}, \mathbf{R}) \leftarrow (p, \vec{\mu}_i, \vec{\sigma}_i, \mathbf{M}_i, \mathbf{R}_i)$
  else
    if $|\vec{\mu}| \ne |\vec{\mu}_i|$ OR $\vec{\sigma} >_{lex} \vec{\sigma}_i$ OR ($\vec{\sigma} = \vec{\sigma}_i$ AND $\vec{\mu}$ is further from $w$ than $\vec{\mu}_i$)
    then
      {Previous primes are unlucky}
      $(q, \vec{\mu}, \vec{\sigma}, \mathbf{M}, \mathbf{R}) \leftarrow (1, \vec{0}, \vec{0}, \mathbf{0}, \mathbf{0})$
    end if
    if $q = 1$ OR $(\vec{\mu}, \vec{\sigma}) = (\vec{\mu}_i, \vec{\sigma}_i)$ then
      {Current prime may be lucky}
      $(\mathbf{M}, \mathbf{R}) \leftarrow (\mathrm{CRA}(\mathbf{M}, \mathbf{M}_i, q, p), \mathrm{CRA}(\mathbf{R}, \mathbf{R}_i, q, p))$
      $q \leftarrow qp$
    end if
  end if
end while
```

Example 5.10 *Let*

$$\mathbf{F}(z) = \begin{bmatrix} \mathbf{A}(z) \\ \mathbf{B}(z) \end{bmatrix}$$

with

$$\mathbf{A}(z) = \begin{bmatrix} 3\,z^4+3\,z^3+4\,z^2-2\,z-4 & 3\,z^4+3\,z^2+14\,z+8 \\ z^4+5\,z^3+3\,z^2+3\,z+1 & z^4+7\,z^3+6\,z^2+z+1 \end{bmatrix},$$

$$\mathbf{B}(z) = \begin{bmatrix} z^3+9\,z^2+5\,z+1 & z^3+15\,z^2+19\,z+5 \\ z^5+z^4+2\,z^3+3\,z^2+2\,z+1 & z^5+z^3+7\,z^2+6\,z+1 \end{bmatrix}.$$

Then FFREDUCE2 *over* $\mathbb{Z}$ *gives*

$$d = -2480256 = -2^7 \cdot 3^2 \cdot 2153,$$

$$\vec{\mu} = (5, 4, 3, 2),$$

$$\vec{\sigma} = (1, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16).$$

When $p_1 = 2$*,* FFREDUCE2 *returns* $\vec{\mu}_1 = (3, 3, 2, 2)$*, and when* $p_2 = 3$*,* FFREDUCE2 *returns* $\vec{\mu}_2 = (3, 2, 2, 2)$*. These two primes are unlucky because they both divide* d*. Since* $|\vec{\mu}_1| \neq |\vec{\mu}_2|$*, we simply assume that* $\vec{\mu}_1$ *is unlucky and the corresponding results are discarded. The prime* $p_3 = 5$ *is lucky and the previous results are discarded. However, for* $p_4 = 7$ *we get* $\vec{\mu}_4 = (4, 3, 3, 2)$ *and so* 7 *is unlucky as* $|\vec{\mu}_4| \neq |\vec{\mu}|$*. Since* $|\vec{\mu}_3| \neq |\vec{\mu}_4|$*, the previous results are also discarded. The primes* $p_i = 11, 13, \ldots, 37$

are all lucky, and we can terminate the algorithm because the reconstructed results satisfy $\mathbf{U}(z) \cdot \mathbf{F}(z) = \mathbf{T}(z)$. □

For the complexity analysis of ModReduce, we assume that Garner's algorithm has complexity $\mathcal{O}(\text{size}(a)^2)$ for reconstructing the final result of a. These are satisfied when $\mathbb{D} = \mathbb{Z}$ and $\mathbb{D} = \mathbb{Q}_R[x]$ [33].

Theorem 5.11 *Let* $\mathbb{D} = \mathbb{Z}$ *and* K *be a bound on the size of the coefficients appearing in* $\mathbf{F}(z)$. *The worst case bit complexity of* ModReduce *is* $\mathcal{O}((m+s)m^4s^3N^3K^2)$.

Proof. If we apply the same reasoning as Theorem 3.23 and Theorem 4.6 while assuming all arithmetic operations can be done in $\mathcal{O}(K^2)$ time, we see that the bit complexity of FFReduce2 in $\mathbb{Z}_p$ is $\mathcal{O}((m+s)m^2 \min(m,s)^2 N^2 K^2)$, where p is a prime of size approximately K. Moreover, we can compute $\phi_p(\mathbf{F}(z))$ in $\mathcal{O}(msNK^2)$ bit operations. From the remark after Lemma 5.9, we need $\mathcal{O}(m \min(m,s)(mN+1))$ primes, so that the total cost of all invocations of FFReduce2 is $\mathcal{O}((m+s)m^4 \min(m,s)^3 N^3 K^2)$.

Finally, we see from Lemma 3.22 that each coefficient in $\mathbf{M}(z)$ and $\mathbf{R}(z)$ can be reconstructed in $\mathcal{O}((mNsK)^2)$ by the CRA algorithm (over all iterations). Since there are potentially $\mathcal{O}(m^2 N \min(m,s))$ nonzero coefficients in $\mathbf{M}(z)$ and $\mathcal{O}(msN)$ nonzero coefficients in $\mathbf{R}(z)$, it follows that the reconstruction process has complexity $\mathcal{O}((m+s)m^3N^3 \min(m,s)s^2K^2)$. Combining the two parts gives the desired result. □

From the remark after Lemma 5.9, we can choose the primes in such a way that it is unlikely to require more than $\mathcal{O}(\min(m,s)(mN+1))$ to obtain enough lucky primes. Therefore, we have the following expected bit complexity.

Corollary 5.12 *Let $\mathbb{D} = \mathbb{Z}$ and K be a bound on the size of the coefficients appearing in $\mathbf{F}(z)$. The expected bit complexity of* ModReduce *is $\mathcal{O}((m+s)m^3s^3N^3K^2)$.* $\square$

Compared to the complexity given in Theorem 4.6 for FFReduce, we see that the modular algorithm is an order of magnitude faster in each of the three parameters m, s, and N. When $\mathbb{D} = \mathbb{Q}_R[x]$, we have

Theorem 5.13 *Let $\mathbb{D} = \mathbb{Q}_R[x]$ and K be a bound on the degree of the coefficients appearing in $\mathbf{F}(z)$. Then* ModReduce *requires $\mathcal{O}((m+s)m^4s^3N^3K^2)$ operations in $\mathbb{Q}_R$ in the worst case. The expected complexity is $\mathcal{O}((m+s)m^3s^3N^3K^2)$ operations in $\mathbb{Q}_R$.* $\square$

5.6 Experimental Results

Both FFReduce and ModReduce have been implemented in Maple for $\mathbb{D} = \mathbb{Z}$. Since Maple uses base-10 arithmetic, the size of a coefficient is the number of decimal digits required to represent it. Instead of using primes of size K, we chose primes that are half the machine word size, so that modular arithmetic can be performed efficiently. We used the `modp1` representation for polynomials for the efficient implementation of the modular algorithm. We also implemented early termination in ModReduce.

We now present some experimental results to support the complexity analysis

s	N	K	Size	FFREDUCE (s)	MODREDUCE (s)	Ratio
1	20	4	302	1.53	3.93	0.39
1	40	4	599	14.21	17.63	0.81
1	60	4	905	61.36	44.82	1.37
1	20	9	699	5.12	9.82	0.52
1	40	9	1428	86.58	47.70	1.82
1	60	9	1533	114.58	60.04	1.91
2	20	4	622	28.16	79.10	0.36
2	40	4	1210	413.55	342.67	1.21
2	60	4	1680	1511.62	940.47	1.61
2	20	9	1507	158.79	231.82	0.68
2	40	9	2751	1664.87	827.86	1.99
2	60	9	3933	6432.79	2191.53	2.94

Table 5.1: Comparison of FFREDUCE and MODREDUCE for various values of s, N, and K. Also shown is the size (in number of decimal digits) of the largest coefficient in the result.

given in Theorem 5.11[2]. We note the multiplications in Maple are subquadratic, so that the actual improvement may not be accurately predicted by Theorem 5.11. The input were chosen to compute a row-reduced GCRD of two $s \times s$ polynomial matrices (Section 4.5) $\mathbf{A}(z)$ and $\mathbf{B}(z)$, so that $m = 2s$. In the first set of experiments, we chose various values of s, N, and K, and generated $\mathbf{A}(z)$ and $\mathbf{B}(z)$ randomly. In these cases, $\mathbf{A}(z)$ and $\mathbf{B}(z)$ are usually right coprime. The experimental results are presented in Table 5.1. In the second set of experiments, we randomly generated $\mathbf{C}(z)$ of degree d. We generated $\mathbf{A}(z)$ and $\mathbf{B}(z)$ by multiplying $\mathbf{C}(z)$ on the right to random polynomial matrices of degree $N - d$. This allows us to have some control over the degree of the computed GCRD[3]. The results from the second set

[2]The experiments were performed on an Intel Pentium III 650 MHz with 256MB of RAM.

[3]To precisely control the degree, we would have to control the degree of $\det \mathbf{C}(z)$.

d	Size	FFReduce (s)	ModReduce (s)	Ratio
1	3858	6895.10	2315.51	2.98
10	3213	3381.31	1432.72	2.36
20	2719	2200.21	1130.49	1.95
40	1425	212.83	256.36	0.83

Table 5.2: Comparison of FFReduce and ModReduce for various values of d with $s = 2$, $N = 60$, and $K = 9$. Also shown is the size (in number of decimal digits) of the largest coefficient in the result.

of experiments are presented in Table 5.2. We see that as s, N, and K increases, the advantage of the modular algorithm over the fraction-free algorithm becomes clear. Moreover, this advantage is also apparent when the degree of $\mathbf{C}(z)$ is small. These conditions encourage coefficient growth, and so the modular algorithm is significantly better in these cases. For small values of these parameters, the modular algorithm is slower because of the additional overhead performed.

5.7 Images Under Unlucky Homomorphisms

While unlucky homomorphisms are rarely encountered, it is still wasteful to discard the computed results. In this section, we look at the special case when $|\vec{\mu}_i| \neq |\vec{\mu}|$. Here we can sometimes determine the image $\phi_i(\mathbf{M}(z))$ and $\phi_i(\mathbf{R}(z))$ even if ϕ_i is unlucky. This is based on same technique given by Cabay [22, 48], which states that if the $n \times n$ coefficient matrix A for a system of linear equations over $\mathbb{Q}_{\mathbb{D}}$ has rank n over $\mathbb{Q}_{\mathbb{D}}$ but has rank less than $n - 1$ over $\mathbb{D}/I$, then the image of the Cramer solution in $\mathbb{D}/I$ is zero; if the rank over $\mathbb{D}/I$ is exactly $n-1$, then additional calculations are required to compute the image.

Suppose that $|\vec{\mu}_i| < |\vec{\mu}|$, so that ϕ_i is unlucky. We will further assume that $\vec{\omega} \leq \vec{\omega}_i$. We recall from (3.10) and (3.11) that the rank of the coefficient matrix $K_i^*(\vec{\mu}_i - \vec{e}, \vec{\omega}_i)$ over $\mathbb{D}/I$ is $|\vec{\mu}_i|$. Now Definition 2.16 implies that rank $\phi_i(K^*(\vec{\mu} - \vec{e}, \vec{\omega})) \leq |\vec{\mu}_i|$ over $\mathbb{D}/I$. From the previous discussion, we have the following.

Theorem 5.14 *Let $\vec{\mu}_i$ be the degree of the Mahler system computed by* FFREDUCE2 *at order $\vec{\omega}_i$ over $\mathbb{D}/I$. If $|\vec{\mu}_i| < |\vec{\mu}| - 1$ and $\vec{\omega} \leq \vec{\omega}_i$, then $\phi_i(\mathbf{M}(z)) = \mathbf{0}$ and $\phi_i(\mathbf{R}(z)) = \mathbf{0}$.* □

On the other hand, it is also easy to compute $\phi_i(\mathbf{M}(z))$ and $\phi_i(\mathbf{R}(z))$ if $|\vec{\mu}_i| = |\vec{\mu}| - 1$ and $\vec{\mu} = \vec{\mu}_i + \vec{e}_\pi$ for some $1 \leq \pi \leq m$. In fact, we may simply perform FFREDUCE2 from the intermediate results $\mathbf{M}_i(z)$ and $\mathbf{R}_i(z)$ corresponding to order $\vec{\omega}$ for one additional iteration, using row π as the pivot. Since the known divisor d_k is nonzero in the previous step, the division is valid and mirrors the computation performed over $\mathbb{D}$.

However, if $\vec{\mu}_i \not\leq \vec{\mu}$, it is not clear how to compute $\phi_i(\mathbf{M}(z))$ and $\phi_i(\mathbf{R}(z))$ when ϕ_i is unlucky, because the computed results $\mathbf{M}_i(z)$ and $\mathbf{R}_i(z)$ correspond to systems of linear equations with coefficient matrix $K(\vec{\mu}_i - \vec{e}, \vec{\omega})$, which is not a submatrix of $K(\vec{\mu} - \vec{e}, \vec{\omega})$.

5.8 Coprime Polynomial Matrices

It is well known that there exists a fast modular algorithm to decide if two polynomials are relatively prime. Namely, if $\deg \phi(a(z)) = \deg a(z)$, $\deg \phi(b(z)) = \deg b(z)$, and $g'(z)$ is the GCD of $\phi(a(z))$ and $\phi(b(z))$ in $(\mathbb{D}/I)[z]$, then $\deg \mathrm{GCD}(a(z), b(z)) \leq$

$\deg g'(z)$ [36]. In particular, if $\deg g'(z) = 0$, it follows that $\deg \mathrm{GCD}(a(z), b(z)) = 0$ and so $a(z)$ and $b(z)$ are relatively prime. In this section, we examine the possibility of an efficient test for relatively primeness of two polynomial matrices.

The observation above can be extended to the case of polynomial matrices. For simplicity we assume that the input matrices are square and nonsingular.

Theorem 5.15 *Let* $\mathbf{A}(z) \in \mathbb{D}[z]^{s\times s}$, $\mathbf{B}(z) \in \mathbb{D}[z]^{s\times s}$. *Let* $\mathbf{G}(z)$ *be a GCRD of* $\mathbf{A}(z)$ *and* $\mathbf{B}(z)$ *over* $\mathbb{Q}_{\mathbb{D}}$, *and* $\mathbf{G}'(z)$ *be a GCRD of* $\phi(\mathbf{A}(z))$ *and* $\phi(\mathbf{B}(z))$ *over* $\mathbb{D}/I$. *If* $\deg \phi(\det \mathbf{A}(z)) = \deg\det \mathbf{A}(z)$ *or* $\deg \phi(\det \mathbf{B}(z)) = \deg\det \mathbf{B}(z)$, *then*

$$\deg\det \mathbf{G}(z) \leq \deg\det \mathbf{G}'(z).$$

Furthermore, if $\deg\det \mathbf{G}'(z) = 0$ *then* $\mathbf{A}(z)$ *and* $\mathbf{B}(z)$ *are right coprime.*

Proof. Suppose that $\mathbf{A}(z) = \mathbf{Q}_1(z) \cdot \mathbf{G}(z)$ and $\mathbf{B}(z) = \mathbf{Q}_2(z) \cdot \mathbf{G}(z)$ for some $\mathbf{Q}_1(z)$ and $\mathbf{Q}_2(z)$ with entries in $\mathbb{Q}_{\mathbb{D}}[z]$. Examining these equations over $\mathbb{D}/I$ shows that $\phi(\mathbf{G}(z))$ is a common right divisor of $\phi(\mathbf{A}(z))$ and $\phi(\mathbf{B}(z))$. Thus, there exists $\mathbf{Q}(z)$ with entries in $(\mathbb{D}/I)[z]$ such that $\mathbf{G}'(z) = \mathbf{Q}(z) \cdot \phi(\mathbf{G}(z))$ in $(\mathbb{D}/I)[z]$.

Now, $\det \mathbf{A}(z) = (\det \mathbf{Q}_1(z))(\det \mathbf{G}(z))$ and $\det \mathbf{B}(z) = (\det \mathbf{Q}_2(z))(\det \mathbf{G}(z))$. Hence, $\deg \phi(\det \mathbf{A}(z)) = \deg\det \mathbf{A}(z)$ or $\deg \phi(\det \mathbf{B}(z)) = \deg\det \mathbf{B}(z)$ implies that $\deg\det \mathbf{G}(z) = \deg \phi(\det \mathbf{G}(z))$. Therefore,

$$\deg\det \mathbf{G}(z) = \deg \phi(\det \mathbf{G}(z)) \leq \deg \phi(\det \mathbf{G}(z)) + \deg\det \mathbf{Q}(z) = \deg\det \mathbf{G}'(z)$$

as required, where $\det \mathbf{G}(z)$ is computed in $\mathbb{Q}_{\mathbb{D}}[z]$ and $\det \mathbf{Q}(z)$ and $\det \mathbf{G}'(z)$ are computed in $(\mathbb{D}/I)[z]$. Finally, if $\deg\det \mathbf{G}'(z) = 0$, then $\mathbf{G}(z)$ must be unimodular so that $\mathbf{A}(z)$ and $\mathbf{B}(z)$ are right coprime. $\square$

In order to use Theorem 5.15 for a more efficient modular algorithm to detect coprimeness, we first have to compute $\deg\det\mathbf{A}(z)$ and $\deg\det\mathbf{B}(z)$. This can be done by computing row-reduced forms $\mathbf{A}'(z)$ and $\mathbf{B}'(z)$ of $\mathbf{A}(z)$ and $\mathbf{B}(z)$, respectively. The second part of the algorithm then computes a row-reduced form of $[\mathbf{A}'(z)^T,\ \mathbf{B}'(z)^T]^T$. While Theorem 5.15 may apply to the second part to detect coprimeness of the algorithm quickly, it is not clear how to speed up the first part. If $\mathbf{A}(z)$ and $\mathbf{B}(z)$ are already row-reduced, however, this step can be made more efficient as well. The following theorem gives a more efficient test to detect whether a polynomial matrix is row-reduced.

Theorem 5.16 *Let $\mathbf{T}(z)$ be a row-reduced form of $\phi(\mathbf{A}(z))$ over $\mathbb{D}/I$. If $|\text{rdeg } \mathbf{A}(z)| = |\text{rdeg } \mathbf{T}(z)|$, then $\mathbf{A}(z)$ is row-reduced over $\mathbb{Q}_\mathbb{D}$.*

Proof. Let $\mathbf{T}_2(z)$ be a row-reduced form of $\mathbf{A}(z)$ over $\mathbb{Q}_\mathbb{D}$. Then $|\text{rdeg } \mathbf{T}_2(z)| = \deg\det\mathbf{A}(z)$. Therefore,

$$|\text{rdeg } \mathbf{T}(z)| = \deg\phi(\det\mathbf{A}(z)) \leq \deg\det\mathbf{A}(z) = |\text{rdeg } \mathbf{T}_2(z)| \leq |\text{rdeg } \mathbf{A}(z)|.$$

It follows that if $|\text{rdeg } \mathbf{T}(z)| = |\text{rdeg } \mathbf{A}(z)|$, then $|\text{rdeg } \mathbf{T}_2(z)| = |\text{rdeg } \mathbf{A}(z)|$ and so $\mathbf{A}(z)$ is row-reduced over $\mathbb{Q}_\mathbb{D}$. $\square$

Chapter 6

A Modular Algorithm for Popov Form for Polynomial Matrices

In this chapter we give a modular algorithm for computing the Popov form for polynomial matrices. Our algorithm uses the ModReduce algorithm given in Algorithm 5.1 as a subroutine. As in the previous chapter, we will assume that $\mathbb{D} = \mathbb{Z}$ or $\mathbb{D} = \mathbb{Q}_R[x]$ in this chapter.

6.1 Issues

As noted in Section 2.3, if $\mathbf{P}(z)$ is the Popov form of any input polynomial matrix $\mathbf{F}(z) \in \mathbb{D}[z]^{m\times s}$, then $\mathbf{P}(z) \in \mathbb{Q}_{\mathbb{D}}[z]^{m\times s}$ and is unique up to row permutation. Unlike the row-reduced and weak Popov form, we cannot choose $\mathbf{P}(z) \in \mathbb{D}[z]^{m\times s}$. There are two ways to handle this difficulty.

First, we may instead compute $c \in \mathbb{D}$ and $\mathbf{P}'(z) \in \mathbb{D}[z]^{m\times s}$ such that $\mathbf{P}(z) =$

$(1/c) \cdot \mathbf{P}'(z)$ is the desired Popov form. Beckermann, Labahn, and Villard [14, 15] gave an algorithm to compute c, $\mathbf{P}'(z)$, and the corresponding transformation matrix of any polynomial matrix using an indirect algorithm similar to the ones described in Section 2.5.2. The algorithm used for computing the minimal polynomial basis is the FFFG algorithm, which is a special case of the FFReduce algorithm. Applying the same techniques from Chapter 5 automatically gives a modular algorithm to compute the Popov form.

However, we are interested in a direct algorithm based on row operations. Instead, we view the image of each coefficient $a/b \in \mathbb{Q}_{\mathbb{D}}$ as $ab^{-1} \in \mathbb{D}/I_i$ and perform the computations appropriately. We must ensure, of course, that $\phi_i(b) \neq 0$ for any denominator b appearing in the coefficients of $\mathbf{F}(z)$. The result reconstructed by Chinese remaindering gives the image of a/b as $ab^{-1} \in \mathbb{D}/(I_1 \cdots I_k)$. Rational number reconstruction (if $\mathbb{D} = \mathbb{Z}$) or rational function reconstruction (if $\mathbb{D} = \mathbb{Q}_R[x]$) can then be applied to obtain a and b [33].

We also noted in Section 2.3 that the unimodular transformation matrix $\mathbf{U}(z)$ is not unique if $\mathbf{F}(z)$ does not have full row rank. This is easily seen as any row of $\mathbf{U}(z)$ corresponding to an element in the left kernel $\mathcal{N}_{\mathbf{F}(z)}$ can be added to any other row of $\mathbf{U}(z)$ to give another transformation matrix with the same Popov form. In order to apply the modular algorithm to compute the unimodular multiplier, we need to ensure that the results computed under each modular homomorphism correspond to the same result in $\mathbb{Q}_{\mathbb{D}}$. Another issue deals with the detection of unlucky homomorphisms. It is not easy to compare the results computed under two homomorphisms and decide which one, if any, is unlucky. For example, if we compute the GCD of two polynomials $a(z)$ and $b(z)$ by computing the Popov form of

$$\mathbf{F}(z) = \begin{bmatrix} a(z) \\ b(z) \end{bmatrix}, \tag{6.1}$$

the row degree of the resulting Popov form under an unlucky modular homomorphism may be too high [36]. On the other hand, for the polynomial matrix

$$\mathbf{F}(z) = \begin{bmatrix} z^2 & 0 \\ 0 & 3z^2 + z \end{bmatrix}, \tag{6.2}$$

the Popov form computed in $\mathbb{Z}_3$ has row degree that is too low. Thus, it is not possible to compare two results based on row degrees alone.

6.2 Detecting Unlucky Homomorphisms

In this section, we give criteria for detecting unlucky homomorphisms for computing a weak Popov form. These criteria will be the same for detecting unlucky homomorphisms for computing the Popov form. As a side effect, we obtain a modular algorithm for computing a weak Popov form.

While the row degree of the computed Popov form may be either too high or too low, we observe that $\mathbf{F}(z)$ in (6.1) does not have full row rank, while $\mathbf{F}(z)$ in (6.2) has full row rank. In fact, when the input polynomial matrix has full row rank, it is easier to detect unlucky homomorphisms. Our strategy is to first ensure that the modular homomorphism is lucky for the computation of row-reduced form using the results of the previous chapter. This gives us a row-reduced form over $\mathbb{D}/I$, as well as $\vec{\mu}$ and $\vec{\sigma}$ which allow us to detect unlucky homomorphisms without reconstructing the results over $\mathbb{D}$. Once this has been done, the row-reduced form has full row rank and we can compute a weak Popov form.

The ModReduce algorithm can be modified to compute a weak Popov form. First, we have to modify FFReduce2 appropriately to compute a weak Popov form over $\mathbb{D}/I$ as described in Section 4.4. However, the previous definition of lucky homomorphisms is not sufficient to guarantee that the computed result is an image of the same desired result over $\mathbb{D}$. We must also ensure that the pivot indices are identical. In that case, the reconstructed image in $\mathbb{D}$ will also be in weak Popov form.

In the following, we assume that $\mathbf{T}(z)$ is a weak Popov form computed over $\mathbb{D}$, where $\vec{\mu}$ is the degree of the final Mahler system computed and d is its "leading coefficient." Similarly, we define $\mathbf{T}_i(z)$, $\vec{\mu}_i$, and d_i to be the corresponding quantities computed over $\mathbb{D}/I_i$.

Definition 6.1 *Let ϕ_i be a modular homomorphism. Let Π_k be the pivot index of row k of $\mathbf{T}(z)$ over $\mathbb{D}$, and $\Pi_{i,k}$ be the pivot index of row k of $\mathbf{T}(z)$ over $\mathbb{D}/I_i$. Then ϕ_i is* lucky *if $\phi_i(d) \neq 0$, $|\vec{\mu}| = |\vec{\mu}_i|$, and $\Pi_k = \Pi_{i,k}$ for all $k = 1, \ldots, m$. Otherwise, ϕ_i is* unlucky*.* □

For detecting unlucky homomorphisms, we define the vector $\vec{\Pi}$ such that

$$\vec{\Pi}^{(j)} = \begin{cases} k & \text{if } \Pi_k = j \text{ for some } k, \\ 0 & \text{otherwise.} \end{cases}$$

This is well-defined since the pivot indices are unique. Intuitively, $\vec{\Pi}^{(j)}$ gives the index of the row containing a leading element in column j. Similarly, we define $\vec{\Pi}_i$ corresponding to the pivot indices over $\mathbb{D}/I_i$. If we use the pivoting strategy

$$\pi = \min_{1\leq j\leq m}\left\{j : \vec{\mu}^{(j)} = \min_{1\leq k\leq m}\left\{\vec{\mu}^{(k)} : r_k \neq 0\right\}, r_j \neq 0\right\} \tag{6.3}$$

in the FFReduce2 algorithm, the following result is easy to show.

Lemma 6.2 *For any modular homomorphism ϕ_i such that $\phi_i(d) \neq 0$ and $|\vec{\mu}| = |\vec{\mu}_i|$, we have $\vec{\Pi} \leq' \vec{\Pi}_i$ if pivoting strategy (6.3) is used, where $\vec{v} \leq \vec{w}$ if and only if*

(a) $\vec{v} = \vec{w}$, or

(b) there exists j such that $\vec{v}^{(k)} = \vec{w}^{(k)}$ for all $k < j$ and

(i) $\vec{\mu}^{(\vec{v}^{(j)})} < \vec{\mu}^{(\vec{w}^{(j)})}$, or

(ii) $\vec{\mu}^{(\vec{v}^{(j)})} = \vec{\mu}^{(\vec{w}^{(j)})}$ and $\vec{v}^{(j)} < \vec{w}^{(j)}$

Proof. If $\vec{\Pi} >_{lex} \vec{\Pi}_i$, then there exists a column j such that $\vec{\Pi}^{(k)} = \vec{\Pi}_i^{(k)}$ for all $k < j$ and $\vec{\Pi}^{(j)} > \vec{\Pi}_i^{(j)}$. This implies that the first term of the residual in row $\vec{\Pi}_i^{(j)}$ is nonzero in $\mathbb{D}/I_i$ and hence in $\mathbb{D}$, contradicting the minimality of $\vec{\Pi}^{(j)}$ in the pivoting strategy (6.3). □

Combining Theorem 5.7 and Lemma 6.2 allows us to compare the results computed under two modular homomorphisms and detect which is unlucky.

Theorem 6.3 *Suppose ϕ_i and ϕ_j satisfy $\deg \mathbf{F}(z) = \deg \phi_i(\mathbf{F}(z)) = \deg \phi_j(\mathbf{F}(z))$. Then ϕ_i is unlucky if any of the following holds:*

(a) $|\vec{\mu}_i| = |\vec{\mu}_j|$ and $\vec{\sigma}_i >_{lex} \vec{\sigma}_j$.

(b) $|\vec{\mu}_i| = |\vec{\mu}_j|$, $\vec{\sigma}_i = \vec{\sigma}_j$, and $\vec{\mu}_j$ is closer to w than $\vec{\mu}_i$;

(c) $\vec{\mu}_i = \vec{\mu}_j$, $\vec{\sigma}_i = \vec{\sigma}_j$, and $\vec{\Pi}_i >_{lex} \vec{\Pi}_j$.

Furthermore, if $|\vec{\mu}_i| \neq |\vec{\mu}_j|$, then at least one of ϕ_i and ϕ_j is unlucky. $\square$

The remaining parts (and their analysis) of the ModReduce algorithm do not need to be changed. This gives us a modular algorithm for computing a weak Popov form of a polynomial matrix.

Finally, we will show that Definition 6.1 also serves as the definition of lucky homomorphisms for computing the Popov form.

Theorem 6.4 *Suppose that ϕ_i is a lucky homomorphism by Definition 6.1. Let $\mathbf{P}(z)$ be the Popov form of $\mathbf{F}(z)$ over $\mathbb{Q}_{\mathbb{D}}$, and $\mathbf{P}_i(z)$ be the Popov form of $\mathbf{F}(z)$ over $\mathbb{D}/I_i$. Then $\phi_i(\mathbf{P}(z)) = \mathbf{P}_i(z)$.*

Proof. Let $\mathbf{T}(z)$ be the weak Popov form computed by our algorithm over $\mathbb{D}$, and $\mathbf{T}_i(z)$ be the weak Popov form computed by our algorithm over $\mathbb{D}/I_i$. Since ϕ_i is lucky, it follows that $\phi_i(\mathbf{T}(z)) = \mathbf{T}_i(z)$. If we compute the Popov form $\mathbf{P}(z)$ of $\mathbf{T}(z)$ (and hence of $\mathbf{F}(z)$) over $\mathbb{Q}_{\mathbb{D}}$ without row exchanges, the pivot indices remain unchanged. We observe that $\phi_i(\mathbf{P}(z))$ is in Popov form because the entries $\mathbf{P}(z)^{(k,\Pi_k)}$ are monic. Furthermore, since $\Pi_k = \Pi_{i,k}$ for all $k = 1, \ldots, m$, it follows that the leading coefficients of $\mathbf{T}(z)^{(k,\Pi_k)}$ do not vanish under ϕ_i, so the transformation matrix to Popov form is also unimodular over $\mathbb{D}/I_i$. Therefore, $\phi_i(\mathbf{P}(z))$ is the Popov form of $\mathbf{F}(z)$ over $\mathbb{D}/I_i$. On the other hand, computing the Popov form

$\mathbf{P}_i(z)$ of $\mathbf{F}(z)$ from $\mathbf{T}_i(z)$ over $\mathbb{D}/I_i$ in a similar way gives the same pivot indices. Since the Popov form is unique, it follows that $\phi_i(\mathbf{P}(z)) = \mathbf{P}_i(z)$. □

6.3 Minimal Multipliers

Since we also wish to compute the unimodular transformation matrix corresponding to the Popov form, we need to ensure that the transformation matrices computed under different modular homomorphisms correspond to the same result over $\mathbb{Q}_{\mathbb{D}}$. Although the transformation matrix is not unique in general, we can ensure that the transformation matrix is unique if we require its column degree to be minimal [15].

Definition 6.5 *Let $\mathbf{F}(z) \in \mathbb{Q}_{\mathbb{D}}[z]^{m\times s}$ be of rank r. Let $\mathbf{U}(z)$ be a unimodular matrix such that $\mathbf{U}(z)\cdot\mathbf{F}(z) = \mathbf{P}(z)$ where $\mathbf{P}(z)$ is in Popov form with I being the set of $m-r$ row indices of the zero rows, and J being the set of pivot indices. The unimodular matrix $\mathbf{U}(z)$ is called the* minimal multiplier *if*

(a) $\mathbf{U}(z)^{(I,\cdot)}$ *is in Popov form (and hence row-reduced);*

(b) $\mathbf{U}(z)^{(I^c,J)} \cdot \left(\mathbf{U}(z)^{(I,J)}\right)^{-1} = \mathcal{O}(z^{-1})_{z\to\infty}$,

where I^c denotes the complement of I. □

Intuitively, Definition 6.5(b) implies that for the columns indexed by J, the rows in I have been used to reduce the degrees of the entries in the rows in I^c as far as possible. It was also shown by Beckermann, Labahn, and Villard that the minimal multiplier is unique [15, Theorem 3.3].

Theorem 6.6 *The minimal multiplier* $\mathbf{U}(z)$ *is unique (up to row permutation) for any* $\mathbf{F}(z) \in \mathbb{Q}_{\mathbb{D}}[z]^{m \times s}$*, and* $|rdeg\ \mathbf{U}(z)|$ *is minimal among all unimodular multipliers transforming* $\mathbf{F}(z)$ *into Popov form.* □

By computing the minimal multiplier we can ensure that the result computed under each modular homomorphism corresponds to the same desired result in $\mathbb{Q}_{\mathbb{D}}$.

6.4 Computing Homomorphic Images

We now describe how to compute the Popov form $\mathbf{P}_i(z)$ and the minimal multiplier $\mathbf{U}_i(z)$ of $\mathbf{F}(z)$ over $\mathbb{D}/I_i$. As we have already mentioned, we first apply the ModReduce algorithm to compute a weak Popov form $\mathbf{T}_i(z)$. Once this is done, the pivot entry in each nonzero row of $\mathbf{T}_i(z)$ is made monic and is used to reduce the degrees of other rows in the corresponding column. In order to ensure that each row operation do not cancel the progress from the ones already performed, we perform row operations to bring the polynomial matrix into Popov form one row at a time, using as pivots only the rows that have already been transformed into Popov form [50]. The operations are applied to rows of increasing row degrees, and if there is a tie, it is applied to these rows of decreasing pivot indices. To guarantee uniqueness, we will also sort the rows based on the order in which they are processed. This gives us the Popov form $\mathbf{P}_i(z)$ and the corresponding transformation matrix $\mathbf{V}_i(z)$. Finally, the rows of $\mathbf{V}_i(z)$ corresponding to the zero rows of $\mathbf{P}_i(z)$ are already in Popov form except for a scalar multiple by Remark 3.12, and the pivot entries are used to reduce the degrees in the remaining rows. This gives us the minimal multiplier $\mathbf{U}_i(z)$.

6.5 Number of Homomorphic Images Required

In order to examine the complexity of the overall algorithm, we obtain a bound on the size of the coefficients appearing in the Popov form $\mathbf{P}(z)$ and the minimal multiplier $\mathbf{U}(z)$.

Theorem 6.7 *Let K be a bound on the size of the coefficients appearing in $\mathbf{F}(z) \in \mathbb{D}[z]^{m\times s}$, and $a/b \in \mathbb{Q}_{\mathbb{D}}$ be any coefficient appearing in $\mathbf{P}(z)$ and $\mathbf{U}(z)$. Then $size(a), size(b) \leq 2sN(\min(m,s)+1)K$.*

Proof. We see from Theorem 4.2 and Corollary 5.9 of [15] that the FFFG algorithm can be used to compute c, $\mathbf{P}'(z)$, and the corresponding transformation matrix with order $|\vec{\sigma_0}| \leq 2sN(\min(m,s)+1)$. The result now follows from the Hadamard inequality. $\square$

This gives us a bound on the number of homomorphic images required. In the case of $\mathbb{D} = \mathbb{Z}$ the product of lucky primes must exceed $2n^n 2^{2nK}$ where $n = 2sN(\min(m,s)+1)$. In the case of $\mathbb{D} = \mathbb{Q}_R[x]$ we need to have $2(nK+1)$ lucky evaluation points.

6.6 Complexity

To determine the complexity of the algorithm, we first consider the case where $\mathbb{D} = \mathbb{Z}$.

Theorem 6.8 *Let $\mathbb{D} = \mathbb{Z}$, and K be a bound on the size of the coefficients appearing in $\mathbf{F}(z)$. Suppose that the primes chosen have size approximately K and*

$\log n < K$ *with* $n = 2sN(\min(m,s)+1)$*. Then the complexity for computing the Popov form and the minimal multiplier is* $\mathcal{O}(m^3s^2(m+s)\min(m,s)^2N^3K^2)$*.*

Proof. First, note that we need $\mathcal{O}(sN\min(m,s))$ primes with the stated assumptions. To compute the Popov form $\mathbf{P}_i(z)$ of $\mathbf{F}(z)$ over $\mathbb{D}/I_i$, we first compute a weak Popov form using a variation of the FFREDUCE2 algorithm in $\mathcal{O}((m+s)m^2\min(m,s)^2N^2K^2)$ operations. The cost of the transformation to Popov form is $\mathcal{O}(m^3N^2K^2)$ [50, Theorem 7.1]. Therefore, the total cost over all primes is $\mathcal{O}((m+s)m^2s\min(m,s)^3N^3K^2)$.

To compute the minimal multiplier $\mathbf{U}_i(z)$, we note that after the computation of the weak Popov form the corresponding transformation matrix has degree bounded by $\min(m,s)(mN+1)$. For the rows corresponding to the nonzero rows of $\mathbf{P}_i(z)$, the transformation from weak Popov form to the Popov form may increase the degree of the transformation matrix by at most $\deg \mathbf{P}_i(z) \le N$. By [15, Lemma 3.5], the row degree of the multiplier cannot increase while reducing the rows corresponding to the nonzero rows of $\mathbf{P}_i(z)$. Thus, the complexity for computing the minimal multiplier is $\mathcal{O}(m^3(\min(m,s)(mN+1)+N)K^2) = \mathcal{O}(m^4\min(m,s)NK^2)$. Over all primes the cost is $\mathcal{O}(m^4s\min(m,s)^2N^2K^2)$.

Finally, the reconstruction for Chinese remaindering and rational reconstruction can both be done in $\mathcal{O}(s^2\min(m,s)^2N^2K^2)$ operations for each coefficient [33]. Since there are at most $\mathcal{O}(msN)$ nonzero coefficients in $\mathbf{P}(z)$ and $\mathcal{O}(m^3(m+s)N)$ nonzero coefficients in $\mathbf{U}(z)$ (Theorem 5.1(a) with $\vec{a} = \vec{b} = 0$ of [15]), the reconstruction process has complexity $\mathcal{O}(m^3s^2(m+s)\min(m,s)^2N^3K^2)$. $\square$

Similarly, we obtain the complexity for the case of $\mathbb{D} = \mathbb{Q}_R[x]$.

Theorem 6.9 *Let $\mathbb{D} = \mathbb{Q}_R[x]$ and K be a bound on the degree of the coefficients appearing in $\mathbf{F}(z)$. Then the Popov form and the minimal multiplier can be computed in $\mathcal{O}(m^3 s^2 (m+s) \min(m,s)^2 N^3 K^2)$ operations in $\mathbb{Q}_R$.* $\square$

This compares favorably to the complexity of the indirect fraction-free algorithm of Beckermann, Labahn, and Villard [14] of $\mathcal{O}(m^4 s^5 N^4 K^2)$.

Chapter 7

Conclusion and Future Work

7.1 Summary of Contribution

In this thesis, we first gave the FFReduce algorithm which allows us to determine the rank and a row-reduced basis of the left nullspace of a matrix of Ore polynomials in a fraction-free way. By expressing row reduction as a linear algebra problem, we were able to obtain bounds on the size of the intermediate results and gave a complexity analysis of the algorithm. In the case of matrices of shift polynomials, the FFReduce algorithm satisfies additional properties and it can be used to compute rational solutions of linear functional systems, as well as a row-reduced form or a weak Popov form of such matrices in a fraction-free way. The unimodular transformation matrices corresponding to the row-reduced form and the weak Popov form are also computed. We also showed that our approach can be considered as a generalization of the subresultant to the matrix case, and can be used to compute the GCRD and LCLM of matrices of shift polynomials.

We then examined the computation of the desired normal forms in the special case of polynomial matrices using modular homomorphisms and Chinese remaindering. Using the linear algebra formulation for the fraction-free algorithm, we obtained the criteria for lucky homomorphisms for computing a row-reduced form, a weak Popov form, and the Popov form of polynomial matrices. We also studied efficient tests for coprimeness and row-reducedness of polynomial matrices.

Our work can be viewed as an extension of the FFFG algorithm by Beckermann and Labahn [12] in a number of ways. The FFFG algorithm is extended to the case of Ore polynomials, where we had to modify the fraction-free recursion and the termination criteria. We also extended the FFFG algorithm to compute the weak Popov form in the case of shift polynomials. Finally, we used the FFFG algorithm as a basis to design a modular algorithm for polynomial matrices.

7.2 Future Research Directions

Extension of the subresultant algorithms. The FFReduce algorithm is designed to eliminate low order coefficients. In the case of matrices of shift polynomials, we can easily reverse the coefficients to eliminate the leading coefficients as well. Unfortunately, this approach does not work in the general case of matrices of Ore polynomials. In addition, our algorithm makes use of the Mahler system in the prediction of known factors. This forces us to also compute the transformation matrix whose coefficients may be much larger than those in the residual. On the other hand, the subresultant algorithms for both polynomials and Ore polynomials eliminate leading coefficients directly and predict known divisors without computing the corresponding cofactors [19, 20, 28, 44, 45]. It may be possible to extend these subresultant algorithms to matrices of Ore polynomials. To our knowledge this has not been studied even in the case of polynomial matrices.

Direct fraction-free algorithm for the Popov form. The fraction-free FFReduce algorithm cannot be used to compute the Popov form, even in the case of polynomial matrices. While fraction-free Gaussian elimination can be applied to make the leading row coefficient diagonal (up to row permutation) by back-substitution [6], we do not know how to apply it to make the leading column coefficient diagonal as well. As noted before, an indirect fraction-free algorithm for the Popov form already exists [14, 15].

Different linear algebra formulation. It may be possible to formulate the problem in a different way instead of using the notion of order basis. Since the order increases after each iteration in the algorithm, the pivot row is forced to be modified. This causes the coefficients to grow, and the module generated by the rows of the result is no longer the same as the module generated by the original input matrix. On the other hand, standard row reduction algorithms for computing these normal forms do not modify the pivot row in each step. A different linear algebra formulation of row reduction may eliminate these problems.

Row-reduced and weak Popov forms with smaller coefficients. While the coefficient growth in our fraction-free algorithms is controlled, the coefficients can be large because they are defined as determinants of large matrices. In fact, the algorithms compute exactly the same intermediate results as if the fraction-free Gaussian elimination of Bareiss [6] was applied to the striped Krylov matrix. While this elimination produces the smallest possible coefficients in general, in most cases a large content in each row remains and can be removed. If we perform the Gaussian elimination in a different way, we may obtain smaller coefficients. For example, we can remove the content of each row after removing the known divisor. If we keep track of the content removed from each row we may be able to continue predicting the known divisor.

The FFReduce algorithm was used as a basis for the modular algorithms. Consequently, the results computed by the modular algorithms were the same as those computed by the FFReduce algorithm. We have experimented with making results smaller, for example, by dividing the results by the multigradient d and use rational reconstruction to obtain the final result. Unfortunately, we have not observed significant improvements due to the size of the coefficients in the transformation matrix. It would be interesting to investigate other ways to detect unlucky homomorphisms and solve the normalization problem so that the reconstructed normal forms over $\mathbb{D}$ have smaller coefficients.

Early termination for modular algorithms. Our modular algorithms terminate early when the reconstructed result has not changed for additional homomorphisms. In order to ensure that the correct result is obtained, we check our result by comparing $\mathbf{U}(z) \cdot \mathbf{F}(z) = \mathbf{T}(z)$. On the other hand, Cabay showed that it is possible to ensure the correctness of the reconstructed result without additional checks if the reconstructed result is not updated for a sufficient number of steps [22]. Such a termination criterion is especially important if we have an algorithm that computes the normal form without the corresponding transformation matrix. Without it we would have to perform "trial division" as in the case of modular polynomial GCD algorithms. It is not completely clear how Cabay's technique can be applied to our algorithms. For row-reduced form and weak Popov form, it can be used to ensure that the reconstructed result is the correct Mahler system of a particular row degree, it cannot be used to ensure that the Mahler system of the correct row degree is reached. In other words, we cannot be sure that the number of zero rows in the residual is correct. However, if we perform the algorithm over $\mathbb{D}/I$ to obtain an order basis of order $(mN + 1) \cdot \vec{e}$, then Theorem 3.20 and the fact that the Mahler system is correct ensure that the number of zero rows in the residual is also correct. This requires more iterations to be performed for each homomorphic image, and it is not clear if the trade-off is advantageous. Also, our modular algorithm for the Popov form is not formulated as the solutions of a linear algebra system, so Cabay's technique is not directly applicable.

Modular algorithm for matrices of Ore or shift polynomials. We described modular algorithms only for polynomial matrices. If we restrict to the case of $\mathbb{D} = \mathbb{Z}[n]$, we may also use our modular algorithms for the modular homomorphisms $\phi_p : \mathbb{Z}[n][Z; \sigma, \delta] \rightarrow \mathbb{Z}_p[n][Z; \sigma, \delta]$. As shown by Li and Nemes [44, 47], the evaluation homomorphisms $\phi_{n-\alpha} : \mathbb{Z}[n][Z;\sigma,\delta] \rightarrow \mathbb{Z}[Z;\sigma,\delta]$ are not an Ore ring homomorphisms. Therefore, the same approach cannot be applied directly in this case. Li and Nemes solved the problem by performing row reductions on the image of the Sylvester matrix under $\phi_{n-\alpha}$. Since our algorithm is based on row reductions on the striped Krylov matrix, we expect that a similar approach would work.

Other methods for controlling coefficient growth. It would also be interesting to examine other ways to control coefficient growth that have been applied successfully to the polynomial GCD problem or linear system solving. For example, Hensel lifting allows us to solve the linear systems of equations under a single modular homomorphism and successively lift the results [36]. In the case of polynomial matrices, the heuristic GCD method may allow us to eliminate the variable z by substitution [23]. By choosing a sufficiently large evaluation point, a polynomial matrix in Hermite form corresponds to an integer matrix in Hermite form under such substitutions (up to scalar multiples). However, it is not clear what the corresponding integer normal forms are for the normal forms studied in this thesis because they are defined in terms of leading row coefficients. Such connections between polynomial matrices and integer matrices would be interesting.

Bibliography

[1] S. Abramov. EG-eliminations. *Journal of Difference Equations*, 5:393–433, 1999.

[2] S. Abramov and M. Barkatou. Rational solutions of first order linear difference systems. In *Proceedings of the 1998 International Symposium on Symbolic and Algebraic Computation*, pages 124–131. ACM, 1998.

[3] S. Abramov and M. Bronstein. On solutions of linear functional systems. In *Proceedings of the 2001 International Symposium on Symbolic and Algebraic Computation*, pages 1–6. ACM, 2001.

[4] S. Abramov and M. Bronstein. Linear algebra for skew-polynomial matrices. Technical Report RR-4420, INRIA, 2002.

[5] S. Abramov and M. van Hoeij. A method for the integration of solutions of Ore equations. In *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation*, pages 172–175. ACM, 1997.

[6] E. Bareiss. Sylvester's identity and multistep integer-preserving Gaussian elimination. *Mathematics of Computation*, 22(103):565–578, 1968.

[7] M. Barkatou. On rational solutions of systems of linear differential equations. *Journal of Symbolic Computation*, 28(4/5):547–567, 1999.

[8] S. Barnett. *Matrices in Control Theory: with applications to linear programming*. Van Nostrand Reinhold Company, 1971.

[9] B. Beckermann, H. Cheng, and G. Labahn. Fraction-free row reduction of matrices of Ore polynomials. Technical Report CS-2002-37, School of Computer Science, University of Waterloo, 2002.

[10] B. Beckermann, H. Cheng, and G. Labahn. Fraction-free row reduction of matrices of skew polynomials. In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, pages 8–15. ACM, 2002.

[11] B. Beckermann and G. Labahn. Recursiveness in matrix rational interpolation problems. *Journal of Computational and Applied Mathematics*, 77:5–34, 1997.

[12] B. Beckermann and G. Labahn. Fraction-free computation of matrix rational interpolants and matrix GCDs. *SIAM Journal Matrix Analysis and Applications*, 22(1):114–144, 2000.

[13] B. Beckermann and G. Labahn. On the fraction-free computation of column-reduced matrix polynomials via FFFG. Technical Report ANO436, Laboratoire ANO, University of Lille, 2001. Available at `http://ano.univ-lille1.fr/pub/2001/ano436.ps.Z`.

[14] B. Beckermann, G. Labahn, and G. Villard. Shifted normal forms of polyno-

mial matrices. In *Proceedings of the 1999 International Symposium on Symbolic and Algebraic Computation*, pages 189–196. ACM, 1999.

[15] B. Beckermann, G. Labahn, and G. Villard. Normal forms for general polynomial matrices. Technical Report RR2002-01, ENS Lyon, France, 2002.

[16] Th. G. Beelen, G. J. van den Hurk, and C. Praagman. A new method for computing a column reduced polynomial matrix. *Systems & Control Letters*, 10:217–224, 1988.

[17] R. R. Bitmead, S. Y. Kung, B. D. O. Anderson, and T. Kailath. Greatest common divisors via generalized Sylvester and Bezout matrices. *IEEE Transactions on Automatic Control*, AC–23:1043–1046, 1978.

[18] W. E. Boyce and R. C. DiPrima. *Elementary Differential Equations and Boundary Value Problems.* John Wiley & Sons, Inc., 5th edition, 1992.

[19] W. Brown and J. Traub. On Euclid's algorithm and the theory of subresultants. *Journal of the ACM*, 18(4):505–514, 1971.

[20] W. S. Brown. On Euclid's algorithm and the computation of polynomial greatest common divisors. *Journal of the ACM*, 18(4):478–504, 1971.

[21] A. Bultheel and M. van Barel. A matrix Euclidean algorithm and the matrix Padé approximation problem. In C. Brezinski, editor, *Continued Fractions and Padé Fractions.* Elsevier, North-Holland, 1990.

[22] S. Cabay. Exact solution of linear equations. In *Proceedings of the Second Symposium on Symbolic and Algebraic Manipulation*, pages 392–398, 1971.

[23] B. W. Char, K. O. Geddes, and G. H. Gonnet. GCDHEU: Heuristic polynomial GCD algorithm based on integer GCD computation. *Journal of Symbolic Computation*, 9:31–48, 1989.

[24] H. Cheng and G. Labahn. A modular greatest common divisor algorithm for matrix polynomials. Technical Report CS-2002-04, School of Computer Science, University of Waterloo, 2002.

[25] F. Chyzak and B. Salvy. Non-commutative elimination in Ore algebras proves multivariate holonomic identities. *Journal of Symbolic Computation*, 26(2):187–227, 1998.

[26] B. Codenotti and G. Lotti. A fast algorithm for the division of two polynomial matrices. *IEEE Transactions on Automatic Control*, 34(4):446–448, 1989.

[27] P. M. Cohn. *Free Rings and Their Relations.* Academic Press, 1971.

[28] G. E. Collins. Subresultants and reduced polynomial remainder sequences. *Journal of the ACM*, 14(1):128–142, 1967.

[29] G. E. Collins. The calculation of multivariate polynomial resultants. *Journal of the ACM*, 18(4):515–532, 1971.

[30] D. S. Dummit and R. M. Foote. *Abstract Algebra.* Prentice-Hall, Inc., 1991.

[31] S. H. Friedberg, A. J. Insel, and L. E. Spence. *Linear Algebra.* Prentice-Hall, Inc., 2nd edition, 1989.

[32] H. Garner. The residue number system. *IRE Transactions on Electronic Computers*, EC-8:140–147, 1959.

[33] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 1999.

[34] J. von zur Gathen and S. Hartlieb. Factoring modular polynomials. In *Proceedings of the 1996 International Symposium on Symbolic and Algebra Computation*, pages 10–17. ACM, 1996.

[35] J. von zur Gathen and S. Hartlieb. Factoring modular polynomials. *Journal of Symbolic Computation*, 26(5):583–606, 1998.

[36] K. O. Geddes, S. R. Czapor, and G. Labahn. *Algorithms for computer algebra*. Kluwer Academic Publishers, 1992.

[37] C. Gentle. Computing greatest common divisors of polynomial matrices. Master's thesis, University of Waterloo, 1999.

[38] A. J. Geurts and C. Praagman. Algorithm 7667: A Fortran 77 package for column reduction of polynomial matrices. *ACM Transactions on Mathematical Software*, 23(1):111–129, 1997.

[39] J. L. Hafner and McCurley. Asymptotically fast triangularization of matrices over rings. *SIAM Journal on Computing*, 20(6):1068–1083, 1991.

[40] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

[41] T. Kailath. *Linear Systems*. Prentice-Hall, 1980.

[42] D. E. Knuth. *The Art of Computer Programming*, volume 2. Addison Wesley, 3rd edition, 1998.

[43] V. Kučera. *Discrete Linear Control*. John Wiley & Sons, 1979.

[44] Z. Li. *A Subresultant Theory for Linear Differential, Linear Difference and Ore Polynomials, with Applications*. PhD thesis, RISC-Linz, Johannes Kepler University, Linz, Austria, 1996.

[45] Z. Li. A subresultant theory for Ore polynomials with applications. In *Proceedings of the 1998 International Symposium on Symbolic and Algebraic Computation*, pages 132–139. ACM, 1998.

[46] Z. Li. Private communication, Jun 2003.

[47] Z. Li and I. Nemes. A modular algorithm for computing greatest common right divisors of ore polynomials. In *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation*, pages 282–289. ACM, 1997.

[48] J. D. Lipson. *Elements of Algebra and Algebraic Computing*. Addison-Wesley, 1981.

[49] R. Loos. Generalized polynomial remainder sequences. In *Computer Algebra: Symbolic and Algebraic Computation*, pages 115–137. Springer-Verlag, 1982.

[50] T. Mulders and A. Storjohann. On lattice reduction for polynomial matrices. *Journal of Symbolic Computation*, 35(4):377–401, 2003.

[51] W. H. L. Neven and C. Praagman. Column reduction of polynomial matrices. *Linear Algebra and Its Applications*, 188,189:569–589, 1993.

[52] O. Ore. Theory of non-commutative polynomials. *Annals of Mathematics*, 34:480–508, 1933.

[53] G. Villard. Computing Popov and Hermite forms of polynomial matrices. In *Proceedings of the 1996 International Symposium on Symbolic Algebraic Computation*, pages 250–258. ACM, 1996.

[54] Q. G. Wang and C. H. Zhou. An efficient division algorithm for polynomial matrices. *IEEE Transactions on Automatic Control*, 31(2):165–166, 1986.

[55] S. Y. Zhang. The division of polynomial matrices. *IEEE Transactions on Automatic Control*, 31(1):55–56, 1986.

[56] S. Y. Zhang and C. T. Cheng. An algorithm for the division of two polynomial matrices. *IEEE Transactions on Automatic Control*, 28(2):238–240, 1983.