Computing Column Bases of Polynomial Matrices

Wei Zhou and George Labahn
Cheriton School of Computer Science
University of Waterloo,
Waterloo, Ontario, Canada
{w2zhou,glabahn}@uwaterloo.ca

ABSTRACT

Given a matrix of univariate polynomials over a field $\mathbb{K}$, its columns generate a $\mathbb{K}[x]$-module. We call any basis of this module a column basis of the given matrix. Matrix gcds and matrix normal forms are examples of such module bases. In this paper we present a deterministic algorithm for the computation of a column basis of an $m \times n$ input matrix with $m \leq n$. If s is the average column degree of the input matrix, this algorithm computes a column basis with a cost of $O^{\sim}\left(nm^{\omega-1}s\right)$ field operations in $\mathbb{K}$. Here the soft-O notation is Big-O with log factors removed while ω is the exponent of matrix multiplication. Note that the average column degree s is bounded by the commonly used matrix degree that is also the maximum column degree of the input matrix.

Categories and Subject Descriptors: I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms; F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems

General Terms: Algorithms, Theory

Keywords: Order Basis, Kernel basis, Nullspace basis, Column Basis

1. INTRODUCTION

In this paper, we consider the problem of efficiently computing a column basis of a polynomial matrix $\mathbf{F} \in \mathbb{K}[x]^{m\times n}$ with $n \geq m$. A column basis of $\mathbf{F}$ is a basis for the $\mathbb{K}[x]$-module

$$\{\mathbf{Fp} \mid \mathbf{p} \in \mathbb{K}[x]^n \ \} \ .$$

Such a basis can be represented as a full rank matrix $\mathbf{T} \in \mathbb{K}[x]^{m\times r}$ whose columns are the basis elements. A column basis is not unique and indeed any column basis right multiplied by a unimodular polynomial matrix gives another column basis. As a result, a column basis can have arbitrarily high degree. In this paper, the computed column basis has column degrees bounded by the largest column degrees of the input matrix.

Column bases are fundamental constructions in polynomial matrix algebra. As an example, when the row dimension is one (i.e. $m = 1$), then finding a column basis coincides with finding a greatest common divisor (GCD) of all the polynomials in the matrix. Similarly, the nonzero columns of column reduced forms, Popov normal forms, and Hermite normal forms are all column bases satisfying additional degree constraints. A column reduced form gives a special column basis whose column degrees are the smallest possible, while Popov and Hermite forms are special column reduced or shifted column reduced forms satisfying additional conditions that make them unique. Efficient column basis computation is thus useful for fast computation for such core procedures as determining matrix GCDs [4], column reduced forms [7] and Popov forms [15] of $\mathbf{F}$ with any dimension and rank. Column basis computation also provides a deterministic alternative to randomized lattice compression [11, 14].

Column bases are produced by column reduced, Popov and Hermite forms and considerable work has been done on computing such forms, for example [1, 8, 9, 12, 13]. However most of these existing algorithms require that the input matrices be square nonsingular and so start with existing column bases. It is however pointed out in [12, 13] that randomization can be used to relax the square nonsingular requirement.

To compute a column basis, we know from [3] that any matrix polynomial $\mathbf{F} \in \mathbb{K}[x]^{m\times n}$ can be unimodularly transformed to a column basis by repeatedly working with the leading column coefficient matrices. However this method of computing a column basis can be expensive. Indeed one needs to work with up to $\sum \vec{s}$ such coefficient matrices, which could involve up to $\sum \vec{s}$ polynomial matrix multiplications, where a sum without index denotes the sum of all entries.

In this paper we give a fast, deterministic algorithm for the computation of a column basis for $\mathbf{F}$ having complexity $O^{\sim}\left(nm^{\omega-1}s\right)$ field operations in $\mathbb{K}$ with s being the average column degree of $\mathbf{F}$. Here the soft-O notation is Big-O with log factors removed while ω is the exponent of matrix multiplication. Our algorithm works for both rectangular and non-full column rank matrices. Our method depends on kernel basis computation of $\mathbf{F}$ along with finding a factorization of the input matrix polynomial into a column basis and a left kernel basis of a right kernel basis of $\mathbf{F}$. Finding the right and left kernel basis then makes use of the fast kernel basis and order basis algorithms from [20] and [18, 19], respectively.

The remainder of this paper is as follows. Basic definitions and preliminary results on both kernel and order bases are given in the next section. Section 3 provides the matrix factorization form of our input polynomial matrix that forms the core of our procedure, with a column basis being the left factor, and the right factor is a left kernel basis of a right kernel basis of the input matrix. Section 4 provides an algorithm for fast computation of a left kernel basis making use of order bases computation with unbalanced shift. The column basis algorithm is given in Section 5 with the following section giving details on how the methods can be improved when the number of columns is significantly larger than the number of rows. The paper ends with a conclusion along with topics for future research.

2. PRELIMINARIES

In this paper computational cost is analyzed by bounding the number of arithmetic operations in the coefficient field $\mathbb{K}$ on an algebraic random access machine. We assume the cost of multiplying two polynomial matrices with dimension n and degree d is $O^{\sim}(n^{\omega}d)$ field operations, where the multiplication exponent ω is assumed to satisfy $2 < \omega \leq 3$. We refer to the book by [16] for more details and references about the cost of polynomial multiplication and matrix multiplication.

In this section we first describe the notations used in this paper, and then give the basic definitions and properties of *shifted degree*, *order basis* and *kernel basis* for a matrix of polynomials. These will be the building blocks used in our algorithm.

2.1 Notations

For convenience we adopt the following notations in this paper.

Comparing Unordered Lists For two lists $\vec{a} \in \mathbb{Z}^n$ and $\vec{b} \in \mathbb{Z}^n$, let $\bar{a} = [\bar{a}_1, \ldots, \bar{a}_n] \in \mathbb{Z}^n$ and $\bar{b} = [\bar{b}_1, \ldots, \bar{b}_n] \in \mathbb{Z}^n$ be the lists consists of the entries of $\vec{a}$ and $\vec{b}$ but sorted in increasing order.

$$\begin{cases} \vec{a} \geq \vec{b} & \text{if } \bar{a}_i \geq \bar{b}_i \text{ for all } i \in [1, \ldots n] \\ \vec{a} \leq \vec{b} & \text{if } \bar{a}_i \leq \bar{b}_i \text{ for all } i \in [1, \ldots n] \\ \vec{a} > \vec{b} & \text{if } \vec{a} \geq \vec{b} \text{ and } \bar{a}_j > \bar{b}_j \text{ for at least one } j \in [1, \ldots n] \\ \vec{a} < \vec{b} & \text{if } \vec{a} \leq \vec{b} \text{ and } \bar{a}_j < \bar{b}_j \text{ for at least one } j \in [1, \ldots n]. \end{cases}$$

Uniformly Shift a List For a list $\vec{a} = [a_1, \ldots, a_n] \in \mathbb{Z}^n$ and $c \in \mathbb{Z}$, we write $\vec{a} + c$ to denote $\vec{a} + [c, \ldots, c] = [a_1 + c, \ldots, a_n + c]$, with subtraction handled similarly.

Compare a List with a Integer For $\vec{a} = [a_1, \ldots, a_n] \in \mathbb{Z}^n$ and $c \in \mathbb{Z}$, we write $\vec{a} < c$ to denote $\vec{a} < [c, \ldots, c]$, and similarly for $>, \leq, \geq, =$.

2.2 Shifted Degrees

Our methods depend extensively on the concept of *shifted* degrees of polynomial matrices [5]. For a column vector $\mathbf{p} = [p_1, \ldots, p_n]^T$ of univariate polynomials over a field $\mathbb{K}$, its column degree, denoted by $\operatorname{cdeg} \mathbf{p}$, is the maximum of the degrees of the entries of $\mathbf{p}$, that is,

$$\operatorname{cdeg}\ \mathbf{p} = \max_{1 \leq i \leq n} \deg p_i.$$

The *shifted column degree* generalizes this standard column degree by taking the maximum after shifting the degrees by a given integer vector that is known as a *shift*. More specifically, the shifted column degree of $\mathbf{p}$ with respect to a shift $\vec{s} = [s_1, \ldots, s_n] \in \mathbb{Z}^n$, or the *$\vec{s}$-column degree* of $\mathbf{p}$ is

$$\operatorname{cdeg}_{\vec{s}}\ \mathbf{p} = \max_{1 \leq i \leq n} [\deg p_i + s_i] = \deg(x^{\vec{s}} \cdot \mathbf{p}),$$

where $x^{\vec{s}} = \operatorname{diag}\left(x^{s_1}, x^{s_2}, \ldots, x^{s_n}\right)$. For a matrix $\mathbf{P}$, we use $\operatorname{cdeg} \mathbf{P}$ and $\operatorname{cdeg}_{\vec{s}} \mathbf{P}$ to denote respectively the list of its column degrees and the list of its shifted $\vec{s}$-column degrees. When $\vec{s} = [0, \ldots, 0]$, the shifted column degree specializes to the standard column degree. The shifted row degree of a row vector $\mathbf{q} = [q_1, \ldots, q_n]$ is defined similarly as

$$\operatorname{rdeg}_{\vec{s}} \mathbf{q} = \max_{1 \leq i \leq n} [\deg q_i + s_i] = \deg(\mathbf{q} \cdot x^{\vec{s}}).$$

Shifted degrees have been used previously in polynomial matrix computations and in generalizations of some matrix normal forms [6].

The usefulness of the shifted degrees can be seen from their applications in polynomial matrix computation problems [19, 20]. One of its uses is illustrated by the following lemma from [17, Chapter 2], which can be viewed as a stronger version of the predictable-degree property [10].

LEMMA 2.1. *Let $\mathbf{A} \in \mathbb{K}[x]^{m \times n}$ be a $\vec{u}$-column reduced matrix with no zero columns and with $\operatorname{cdeg}_{\vec{u}} \mathbf{A} = \vec{v}$. Then a matrix $\mathbf{B} \in \mathbb{K}[x]^{n \times k}$ has $\vec{v}$-column degrees $\operatorname{cdeg}_{\vec{v}} \mathbf{B} = \vec{w}$ if and only if $\operatorname{cdeg}_{\vec{u}} (\mathbf{AB}) = \vec{w}$.*

The following lemma from [17, Chapter 2] describes a relationship between shifted column degrees and shifted row degrees.

LEMMA 2.2. *A matrix $\mathbf{A} \in \mathbb{K}[x]^{m \times n}$ has $\vec{u}$-column degrees bounded by $\vec{v}$ if and only if its $-\vec{v}$-row degrees are bounded by $-\vec{u}$.*

Another essential fact needed in our algorithm, also based on the use of the shifted degrees, is the efficient multiplication of matrices with unbalanced degrees [20, Theorem 3.7].

THEOREM 2.3. *Let $\mathbf{A} \in \mathbb{K}[x]^{m \times n}$ with $m \leq n$, $\vec{s} \in \mathbb{Z}^n$ a shift with entries bounding the column degrees of $\mathbf{A}$ and ξ, a bound on the sum of the entries of $\vec{s}$. Let $\mathbf{B} \in \mathbb{K}[x]^{n \times k}$ with $k \in O(m)$ and the sum θ of its $\vec{s}$-column degrees satisfying $\theta \in O(\xi)$. Then we can multiply $\mathbf{A}$ and $\mathbf{B}$ with a cost of $O^{\sim}(n^2 m^{\omega-2} s) \subset O^{\sim}(n^{\omega} s)$, where $s = \xi/n$ is the average of the entries of $\vec{s}$.*

2.3 Order Basis

Let $\mathbb{K}$ be a field, $\mathbf{F} \in \mathbb{K}[[x]]^{m \times n}$ a matrix of power series and $\vec{\sigma} = [\sigma_1, \ldots, \sigma_m]$ a vector of non-negative integers.

DEFINITION 2.4. *A vector of polynomials $\mathbf{p} \in \mathbb{K}[x]^{n \times 1}$ has* order $(\mathbf{F}, \vec{\sigma})$ *(or* order $\vec{\sigma}$ *with respect to* $\mathbf{F}$*) if $\mathbf{F} \cdot \mathbf{p} \equiv \mathbf{0} \bmod x^{\vec{\sigma}}$, that is,*

$$\mathbf{F} \cdot \mathbf{p} = x^{\vec{\sigma}} \mathbf{r}$$

for some $\mathbf{r} \in \mathbb{K}[[x]]^{m \times 1}$. If $\vec{\sigma} = [\sigma, \ldots, \sigma]$ has entries uniformly equal to σ, then we say that $\mathbf{p}$ has order $(\mathbf{F}, \sigma)$. The set of all order $(\mathbf{F}, \vec{\sigma})$ vectors is a free $\mathbb{K}[x]$-module denoted by $\langle (\mathbf{F}, \vec{\sigma}) \rangle$.

An order basis for $\mathbf{F}$ and $\vec{\sigma}$ is simply a basis for the $\mathbb{K}[x]$-module $\langle (\mathbf{F}, \vec{\sigma}) \rangle$. We again represent order bases using matrices, whose columns are the basis elements. We only work

with those order bases having minimal or shifted minimal degrees (also referred to as a reduced order basis in [3]), that is, their column degrees or shifted column degrees are the smallest possible among all bases of the module.

An order basis [2, 3] $\mathbf{P}$ of $\mathbf{F}$ with order $\vec{\sigma}$ and shift $\vec{s}$, or simply an $(\mathbf{F}, \vec{\sigma}, \vec{s})$-basis, is a basis for the module $\langle(\mathbf{F}, \vec{\sigma})\rangle$ having minimal $\vec{s}$-column degrees. If $\vec{\sigma} = [\sigma, \ldots, \sigma]$ is uniform then we simply write $(\mathbf{F}, \sigma, \vec{s})$-basis. The precise definition of an $(\mathbf{F}, \vec{\sigma}, \vec{s})$-basis is as follows.

DEFINITION 2.5. *A polynomial matrix $\mathbf{P}$ is an order basis of $\mathbf{F}$ of order $\vec{\sigma}$ and shift $\vec{s}$, denoted by $(\mathbf{F}, \vec{\sigma}, \vec{s})$-basis, if the following properties hold:*

1. *$\mathbf{P}$ is a nonsingular matrix of dimension n and is $\vec{s}$-column reduced.*
2. *$\mathbf{P}$ has order $(\mathbf{F}, \vec{\sigma})$ (or equivalently, each column of $\mathbf{P}$ is in $\langle(\mathbf{F}, \vec{\sigma})\rangle$).*
3. *Any $\mathbf{q} \in \langle(\mathbf{F}, \vec{\sigma})\rangle$ can be expressed as a linear combination of the columns of $\mathbf{P}$, given by $\mathbf{P}^{-1}\mathbf{q}$.*

In this paper, a $(\mathbf{F}, \vec{\sigma}, \vec{s})$-basis is also called a $(\mathbf{F}, \vec{\sigma}, \vec{s})$-order basis to further distinguish it from the kernel basis notation given in the following subsection.

Note that any pair of $(\mathbf{F}, \vec{\sigma}, \vec{s})$-order bases $\mathbf{P}$ and $\mathbf{Q}$ are column bases of each other and are unimodularly equivalent.

We will need to compute order bases with unbalanced shifts using Algorithm 2 from [18]. This computation can be done efficiently as given by the following result from [20].

THEOREM 2.6. *If $\vec{s}$ satisfies $\vec{s} \leq 0$ and $-\sum \vec{s} \leq m\sigma$, then a $(\mathbf{F}, \sigma, \vec{s})$-basis can be computed with a cost of $O^{\sim}(n^{\omega}d)$ field operations, where $d = m\sigma/n$.*

2.4 Kernel Bases

The kernel of $\mathbf{F} \in \mathbb{K}[x]^{m\times n}$ is the $\mathbb{F}[x]$-module

$$\{\mathbf{p} \in \mathbb{K}[x]^n \mid \mathbf{F}\mathbf{p} = 0\}$$

with a kernel basis of $\mathbf{F}$ being a basis of this module. Kernel bases are closely related to order bases, as can be seen from the following definitions.

DEFINITION 2.7. *Given $\mathbf{F} \in \mathbb{K}[x]^{m\times n}$, a polynomial matrix $\mathbf{N} \in \mathbb{K}[x]^{n\times *}$ is a (right) kernel basis of $\mathbf{F}$ if the following properties hold:*

1. *$\mathbf{N}$ is full-rank.*
2. *$\mathbf{N}$ satisfies $\mathbf{F} \cdot \mathbf{N} = 0$.*
3. *Any $\mathbf{q} \in \mathbb{K}[x]^n$ satisfying $\mathbf{F}\mathbf{q} = 0$ can be expressed as a linear combination of the columns of $\mathbf{N}$, that is, there exists some polynomial vector $\mathbf{p}$ such that $\mathbf{q} = \mathbf{N}\mathbf{p}$.*

Any pair of kernel bases $\mathbf{N}$ and $\mathbf{M}$ of $\mathbf{F}$ are column bases of each other and are unimodularly equivalent.

A $\vec{s}$-minimal kernel basis of $\mathbf{F}$ is just a kernel basis that is $\vec{s}$-column reduced.

DEFINITION 2.8. *Given $\mathbf{F} \in \mathbb{K}[x]^{m\times n}$, a polynomial matrix $\mathbf{N} \in \mathbb{K}[x]^{n\times *}$ is a $\vec{s}$-minimal (right) kernel basis of $\mathbf{F}$ if $\mathbf{N}$ is a kernel basis of $\mathbf{F}$ and $\mathbf{N}$ is $\vec{s}$-column reduced. We also call a $\vec{s}$-minimal (right) kernel basis of $\mathbf{F}$ a $(\mathbf{F}, \vec{s})$-kernel basis.*

In our earlier paper [20], a minimal kernel basis is called a minimal nullspace basis. In this paper, the term kernel basis is now used to emphasize the fact that we compute a basis for a $\mathbb{F}[x]$-module instead of a basis for a $\mathbb{F}(x)$-vector space, since the term nullspace basis usually refers to a basis of some vector space as in [14].

We will need to the following result from [20] to bound the sizes of kernel bases.

THEOREM 2.9. *Suppose $\mathbf{F} \in \mathbb{K}[x]^{m\times n}$ and $\vec{s} \in \mathbb{Z}^n_{\geq 0}$ is a shift with entries bounding the corresponding column degrees of $\mathbf{F}$. Then the sum of the $\vec{s}$-column degrees of any $\vec{s}$-minimal kernel basis of $\mathbf{F}$ is bounded by $\sum \vec{s}$.*

We will also need the following result from [20] to compute kernel bases by rows.

THEOREM 2.10. *Let $\mathbf{G} = \left[\mathbf{G}_1^T, \mathbf{G}_2^T\right]^T \in \mathbb{K}[x]^{m\times n}$ and $\vec{t} \in \mathbb{Z}^n$ a shift vector. If $\mathbf{N}_1$ is a $(\mathbf{G}_1, \vec{t})$-kernel basis with $\vec{t}$-column degrees $\vec{u}$, and $\mathbf{N}_2$ is a $(\mathbf{G}_2\mathbf{N}_1, \vec{u})$-kernel basis with $\vec{u}$-column degrees $\vec{v}$, then $\mathbf{N}_1\mathbf{N}_2$ is a $(\mathbf{G}, \vec{t})$-kernel basis with $\vec{t}$-column degrees $\vec{v}$.*

Also recall the cost of kernel basis computation from [20].

THEOREM 2.11. *Given an input matrix $\mathbf{F} \in \mathbb{K}[x]^{m\times n}$. Let $\vec{s} = \operatorname{cdeg}\mathbf{F}$ and $s = \sum \vec{s}/n$ be the average column degree of $\mathbf{F}$. Then a $(\mathbf{F}, \vec{s})$-kernel basis can be computed with a cost of $O^{\sim}(n^{\omega}s)$ field operations.*

3. COLUMN BASIS VIA FACTORIZATION

In this section we reduce the problem of determining a column basis of a polynomial matrix into three separate processes. For this reduction it turns out to be useful to look at following relationship between column basis, kernel basis, and unimodular matrices.

LEMMA 3.1. *Let $\mathbf{F} \in \mathbb{K}[x]^{m\times n}$ and suppose $\mathbf{U} \in \mathbb{K}[x]^{n\times n}$ is a unimodular matrix such that $\mathbf{F}\mathbf{U} = [0, \mathbf{T}]$ with $\mathbf{T}$ of full column rank. Partition $\mathbf{U} = [\mathbf{U}_L, \mathbf{U}_R]$ such that $\mathbf{F} \cdot \mathbf{U}_L = 0$ and $\mathbf{F}\mathbf{U}_R = \mathbf{T}$. Then*

1. *$\mathbf{U}_L$ is a kernel basis of $\mathbf{F}$ and $\mathbf{T}$ is a column basis of $\mathbf{F}$.*
2. *If $\mathbf{N}$ is any other kernel basis of $\mathbf{F}$, then $\mathbf{U}^* = [\mathbf{N}, \mathbf{U}_R]$ is also unimodular and also unimodularly transforms $\mathbf{F}$ to $[0, \mathbf{T}]$.*

PROOF. Since $\mathbf{F}$ and $[0, \mathbf{T}]$ are unimodularly equivalent with $\mathbf{T}$ having full column rank we have that $\mathbf{T}$ is a column basis of $\mathbf{F}$. It remains to show that $\mathbf{U}_L$ is a kernel basis of $\mathbf{F}$. Since $\mathbf{F}\mathbf{U}_L = 0$, $\mathbf{U}_L$ is generated by any kernel basis $\mathbf{N}$, that is, $\mathbf{U}_L = \mathbf{N}\mathbf{C}$ for some polynomial matrix $\mathbf{C}$. Let r be the rank of $\mathbf{F}$, which is also the column dimension of $\mathbf{T}$ and $\mathbf{U}_R$. Then both $\mathbf{N}$ and $\mathbf{U}_L$ have column dimension $n - r$. Hence $\mathbf{C}$ is a square $(n - r) \times (n - r)$ matrix. The unimodular matrix $\mathbf{U}$ can be factored as

$$\mathbf{U} = [\mathbf{N}\mathbf{C}, \mathbf{U}_R] = [\mathbf{N}, \mathbf{U}_R] \begin{bmatrix} \mathbf{C} & 0 \\ 0 & I \end{bmatrix},$$

implying that both factors $[\mathbf{N}, \mathbf{U}_R]$ and $\begin{bmatrix} \mathbf{C} & 0 \\ 0 & I \end{bmatrix}$ are unimodular. Therefore, $\mathbf{C}$ is unimodular and $\mathbf{U}_L = \mathbf{N}\mathbf{C}$ is also a kernel basis. Notice that the unimodular matrix $[\mathbf{N}, \mathbf{U}_R]$ also transforms $\mathbf{F}$ to $[0, \mathbf{T}]$. □

REMARK 3.2. *It is interesting to see what Lemma 3.1 implies in the case of unimodular matrices. Let* $\mathbf{U} \in \mathbb{K}[x]^{n\times n}$ *be a unimodular matrix with inverse* $\mathbf{V}$*, which, for a given* k*, are partitioned as* $\mathbf{U} = [\mathbf{U}_L, \mathbf{U}_R]$ *and* $\mathbf{V} = \begin{bmatrix}\mathbf{V}_U \\ \mathbf{V}_D\end{bmatrix}$ *with* $\mathbf{U}_L \in \mathbb{K}[x]^{n\times k}$ *and* $\mathbf{V}_U \in \mathbb{K}[x]^{k\times n}$*. Since* $\mathbf{U}$ *and* $\mathbf{V}$ *are inverses of each other we have the identities*

$$\mathbf{VU} = \begin{bmatrix}\mathbf{V}_U\mathbf{U}_L & \mathbf{V}_U\mathbf{U}_R \\ \mathbf{V}_D\mathbf{U}_L & \mathbf{V}_D\mathbf{U}_R\end{bmatrix} = \begin{bmatrix}I_k & 0 \\ 0 & I_{n-k}\end{bmatrix}. \qquad (1)$$

Lemma 3.1 then gives:

1. I_k *is a column basis of* $\mathbf{V}_U$ *and a row basis of* $\mathbf{U}_L$*,*
2. I_{n-k} *is a column basis of* $\mathbf{V}_D$ *and a row basis of* $\mathbf{U}_R$*,*
3. $\mathbf{V}_D$ *and* $\mathbf{U}_L$ *are kernel bases of each other,*
4. $\mathbf{V}_U$ *and* $\mathbf{U}_R$ *are kernel bases of each other.*

LEMMA 3.3. *Let* $\mathbf{F} \in \mathbb{K}[x]^{m\times n}$ *with rank* r*. Suppose* $\mathbf{N} \in \mathbb{K}[x]^{n\times(n-r)}$ *is a right kernel basis of* $\mathbf{F}$ *and* $\mathbf{G} \in \mathbb{K}[x]^{r\times n}$ *is a left kernel basis of* $\mathbf{N}$*. Then* $\mathbf{F} = \mathbf{T}\cdot\mathbf{G}$ *with* $\mathbf{T} \in \mathbb{K}[x]^{m\times r}$ *a column basis of* $\mathbf{F}$*.*

PROOF. Let $\mathbf{U} = \begin{bmatrix}\mathbf{U}_L, \mathbf{U}_R\end{bmatrix}$ be a unimodular matrix with inverse $\mathbf{V} = \begin{bmatrix}\mathbf{V}_U \\ \mathbf{V}_D\end{bmatrix}$ partitioned as in equation (1) and satisfying $\mathbf{F}\cdot\mathbf{U} = [0, \mathbf{B}]$ with $\mathbf{B} \in \mathbb{K}[x]^{m\times r}$ a column basis of $\mathbf{F}$. Then $\mathbf{F} = [0, \mathbf{B}]\,\mathbf{U}^{-1} = \mathbf{B}\,[0, I]\,\mathbf{V} = \mathbf{B}\mathbf{V}_D$. Since $\mathbf{V}_D$ is a left kernel basis of $\mathbf{U}_L$, any other left kernel basis $\mathbf{G}$ of $\mathbf{U}_L$ is unimodularly equivalent to $\mathbf{V}_D$, that is, $\mathbf{V}_D = \mathbf{W}\cdot\mathbf{G}$ for some unimodular matrix $\mathbf{W}$. Thus $\mathbf{F} = \mathbf{B}\cdot\mathbf{W}\cdot\mathbf{G}$. Then $\mathbf{T} = \mathbf{B}\cdot\mathbf{W}$ is a column basis of $\mathbf{F}$ since it is unimodularly equivalent to the column basis $\mathbf{B}$. □

Lemma 3.3 outlines a procedure for computing a column basis of $\mathbf{F}$ with three main steps. The first step is to compute a right kernel basis $\mathbf{N}$ of $\mathbf{F}$, something which can be efficiently done using the kernel basis algorithm of [20]. The second step, computing a left kernel basis $\mathbf{G}$ for $\mathbf{N}$ and the third step, computing the column basis $\mathbf{T}$ from $\mathbf{F}$ and $\mathbf{G}$, will still require additional work for efficient computation. Note that, while Lemma 3.3 does not require the bases computed to be minimal, working with minimal kernel bases keeps the degrees well controlled, an important consideration for efficient computation.

EXAMPLE 3.4. *Let*

$$\mathbf{F} = \begin{bmatrix} x^2 & x^2 & x+x^2 & 1+x^2 \\ 1+x+x^2 & x^2 & 1+x^2 & 1+x^2 \end{bmatrix}$$

be a matrix over $\mathbb{Z}_2[x]$*. Then the matrix*

$$\mathbf{N} = \begin{bmatrix} x & 1 \\ 1 & x \\ x & 1 \\ 0 & x \end{bmatrix}$$

is a right kernel basis of $\mathbf{F}$ *and the matrix*

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ x & x^2 & 0 & 1+x^2 \end{bmatrix}$$

is a left kernel basis of $\mathbf{N}$*. Finally the matrix*

$$\mathbf{T} = \begin{bmatrix} x+x^2 & 1 \\ 1+x^2 & 1 \end{bmatrix}$$

satisfies $\mathbf{F} = \mathbf{TG}$*, and is a column basis of* $\mathbf{F}$*.* □

4. COMPUTING A RIGHT FACTOR

Let $\mathbf{N}$ be an $(\mathbf{F}, \vec{s})$-kernel basis computed using the existing algorithm from [20]. Consider now the problem of computing a left $-\vec{s}$-minimal kernel basis $\mathbf{G}$ for $\mathbf{N}$, or equivalently, a right $(\mathbf{N}^T, -\vec{s})$-kernel basis $\mathbf{G}^T$. For this problem, the kernel basis algorithm of [20] cannot be applied directly, since the input matrix $\mathbf{N}^T$ has nonuniform row degrees and negative shift. Comparing to the earlier problem of computing a $\vec{s}$-minimal kernel basis $\mathbf{N}$ for $\mathbf{F}$, it is interesting to note that the original output $\mathbf{N}$ now becomes the new input matrix $\mathbf{N}^T$, while the new output matrix $\mathbf{G}$ has size bounded by the size of $\mathbf{F}$. In other words, the new input has degrees that match the original output, while the new output has degrees bounded by the original input. It is therefore reasonable to expect that the new problem can be computed efficiently. However, we need to find some way to work with the more complicated input degree structure. On the other hand, the simpler output degree structure makes it easier to apply order basis computation in order to compute a $(\mathbf{N}^T, -\vec{s})$-kernel basis.

4.1 Kernel Bases via Order Bases

In order to see how order basis computations can be applied here, let us first recall the following result (Lemma 3.3 [20]) on a relationship between order bases and kernel bases.

LEMMA 4.1. *Let* $\mathbf{P} = [\mathbf{P}_L, \mathbf{P}_R]$ *be any* $(\mathbf{F}, \sigma, \vec{s})$*-order basis and* $\mathbf{N} = [\mathbf{N}_L, \mathbf{N}_R]$ *be any* $\vec{s}$*-minimal kernel basis of* $\mathbf{F}$*, where* $\mathbf{P}_L$ *and* $\mathbf{N}_L$ *contain all columns from* $\mathbf{P}$ *and* $\mathbf{N}$*, respectively, whose* $\vec{s}$*-column degrees are less than* σ*. Then* $[\mathbf{P}_L, \mathbf{N}_R]$ *is a* $\vec{s}$*-minimal kernel basis of* $\mathbf{F}$*, and* $[\mathbf{N}_L, \mathbf{P}_R]$ *is a* $(\mathbf{F}, \sigma, \vec{s})$*-order basis.*

It is not difficult to extend this result to the following lemma to accommodate our situation here.

LEMMA 4.2. *Given a matrix* $\mathbf{A} \in \mathbb{K}[x]^{m\times n}$ *and some integer lists* $\vec{u} \in \mathbb{Z}^n$ *and* $\vec{v} \in \mathbb{Z}^m$ *such that* $\operatorname{rdeg}_{\vec{u}}\mathbf{A} \le \vec{v}$*, or equivalently,* $\operatorname{cdeg}_{-\vec{v}}\mathbf{A} \le -\vec{u}$*. Let* $\mathbf{P}$ *be a* $(\mathbf{A}, \vec{v}+1, -\vec{u})$*-order basis and* $\mathbf{Q}$ *be any* $(\mathbf{A}, -\vec{u})$*-kernel basis. Partition* $\mathbf{P} = [\mathbf{P}_L, \mathbf{P}_R]$ *and* $\mathbf{Q} = [\mathbf{Q}_L, \mathbf{Q}_R]$ *where* $\mathbf{P}_L$ *and* $\mathbf{Q}_L$ *contain all the columns from* $\mathbf{P}$ *and* $\mathbf{Q}$*, respectively, whose* $-\vec{u}$*-column degrees are no more than* 0*. Then*

(i) $[\mathbf{P}_L, \mathbf{Q}_R]$ *is an* $(\mathbf{A}, -\vec{u})$*-kernel basis, and*

(ii) $[\mathbf{Q}_L, \mathbf{P}_R]$ *is an* $(\mathbf{A}, \vec{v}+1, -\vec{u})$*-order basis.*

PROOF. We can use the same proof from Lemma 3.3 in [20]. We know $\operatorname{cdeg}_{-\vec{v}}\mathbf{AP}_L \le \operatorname{cdeg}_{-\vec{u}}\mathbf{P}_L \le 0$, or equivalently, $\operatorname{rdeg}\mathbf{AP}_L \le \vec{v}$. However it also has order greater than $\vec{v}$ and hence $\mathbf{AP}_L = 0$. Thus $\mathbf{P}_L$ is generated by the kernel basis $\mathbf{Q}_L$, that is, $\mathbf{P}_L = \mathbf{Q}_L\mathbf{U}$ for some polynomial matrix $\mathbf{U}$. On the other hand, $\mathbf{Q}_L$ certainly has order $(\mathbf{A}, \vec{v}+1)$ and therefore is generated by $\mathbf{P}_L$, that is, $\mathbf{Q}_L = \mathbf{P}_L\mathbf{V}$ for some polynomial matrix $\mathbf{V}$. We now have $\mathbf{P}_L = \mathbf{P}_L\mathbf{VU}$ and $\mathbf{Q}_L = \mathbf{Q}_L\mathbf{UV}$, implying both $\mathbf{U}$ and $\mathbf{V}$ are unimodular. The result then follows from the unimodular equivalence of $\mathbf{P}_L$ and $\mathbf{Q}_L$ and the fact that they are $-\vec{u}$-column reduced. □

With the help of Lemma 4.2 we can return to the problem of efficiently computing a $(\mathbf{N}^T, -\vec{s})$-kernel basis. In fact, we just need to use a special case of Lemma 4.2, where all the elements of the kernel basis have shifted degrees bounded by 0, thereby making the partial kernel basis be a complete kernel basis.

LEMMA 4.3. *Let* $\mathbf{N}$ *be a* $(\mathbf{F}, \vec{s})$*-kernel basis with* $\text{cdeg}_{\vec{s}}\,\mathbf{N} = \vec{b}$*. Then any* $(\mathbf{N}^T, -\vec{s})$*-kernel basis* $\mathbf{G}^T$ *satisfies* $\text{cdeg}_{-\vec{s}}\mathbf{G}^T \leq 0$*. Let* $\mathbf{P} = [\mathbf{P}_L, \mathbf{P}_R]$ *be a* $\left(\mathbf{N}^T, \vec{b}+1, -\vec{s}\right)$*-order basis, where* $\mathbf{P}_L$ *consists of all columns* $\mathbf{p}$ *satisfying* $\text{cdeg}_{-\vec{s}}\,\mathbf{p} \leq 0$*. Then* $\mathbf{P}_L$ *is a* $(\mathbf{N}^T, -\vec{s})$*-kernel basis.*

PROOF. The column dimension of any $(\mathbf{N}^T, -\vec{s})$-kernel basis $\mathbf{G}^T$ equals the rank r of $\mathbf{F}$. Since both $\mathbf{F}$ and $\mathbf{G}$ are in the left kernel of $\mathbf{N}$, we know $\mathbf{F}$ is generated by $\mathbf{G}$, and the $-\vec{s}$-minimality of $\mathbf{G}$ ensures that the $-\vec{s}$-row degrees of $\mathbf{G}$ are bounded by the corresponding r largest $-\vec{s}$-row degrees of $\mathbf{F}$. These are in turn bounded by 0 since $\text{cdeg}\,\mathbf{F} \leq \vec{s}$. Therefore, any $(\mathbf{N}^T, -\vec{s})$-kernel basis $\mathbf{G}^T$ satisfies $\text{cdeg}_{-\vec{s}}\mathbf{G}^T \leq 0$. The result follows from Lemma 4.2. □

While Lemma 4.3 shows that a complete $(\mathbf{N}^T, -\vec{s})$-kernel basis can be computed by computing a $\left(\mathbf{N}^T, \vec{b}+1, -\vec{s}\right)$-order basis, in fact we do not compute such a order basis, as the computational efficiency can be improved by using Theorem 2.10 to compute a $(\mathbf{N}^T, -\vec{s})$-kernel basis by rows. More specifically, we can partition $\mathbf{N}$ into $[\mathbf{N}_1, \mathbf{N}_2]$ with $\vec{s}$-column degrees $\vec{b}_1$, $\vec{b}_2$ respectively, compute a $(\mathbf{N}_1^T, -\vec{s})$-kernel basis $\mathbf{Q}_1$ with $-\vec{s}$-column degrees $-\vec{s}_2$, and then compute a $(\mathbf{N}_2^T\mathbf{Q}_1, -\vec{s}_2)$-kernel basis $\mathbf{Q}_2$, then $\mathbf{Q}_1\mathbf{Q}_2$ is a $(\mathbf{N}^T, -\vec{s})$-kernel basis. In order to compute the kernel bases $\mathbf{Q}_1$ and $\mathbf{Q}_2$, we still use order basis computations but work with subsets of rows rather than the whole matrix $\mathbf{N}^T$. We now need to make sure that the order bases computed from subsets of rows contain these kernel bases.

LEMMA 4.4. *Let* $\mathbf{N}$ *be partitioned as* $[\mathbf{N}_1, \mathbf{N}_2]$*, with* $\vec{s}$*-column degrees* $\vec{b}_1$*,* $\vec{b}_2$*, respectively. Then we have the following:*

1. *A* $\left(\mathbf{N}_1^T, \vec{b}_1+1, -\vec{s}\right)$*-order basis contains a* $(\mathbf{N}_1^T, -\vec{s})$*-kernel basis whose* $-\vec{s}$*-column degrees are bounded by* 0*.*
2. *If* $\mathbf{Q}_1$ *is this* $(\mathbf{N}_1^T, -\vec{s})$*-kernel basis from above and* $-\vec{s}_2 = \text{cdeg}_{-\vec{s}}\mathbf{Q}_1$*, then a* $\left(\mathbf{N}_2^T\mathbf{Q}_1, \vec{b}_2+1, -\vec{s}_2\right)$*-basis contains a* $(\mathbf{N}_2^T\mathbf{Q}_1, -\vec{s}_2)$*-kernel basis,* $\mathbf{Q}_2$*, whose* $-\vec{s}$*-column degrees are bounded by* 0*.*
3. *The product* $\mathbf{Q}_1\mathbf{Q}_2$ *is a* $(\mathbf{N}^T, -\vec{s})$*-kernel basis.*

PROOF. To see that a $\left(\mathbf{N}_1^T, \vec{b}_1+1, -\vec{s}\right)$-basis contains a $(\mathbf{N}_1^T, -\vec{s})$-kernel basis whose $-\vec{s}$-column degrees are bounded by 0, we just need to show that $\text{cdeg}_{-\vec{s}}\bar{\mathbf{Q}}_1 \leq 0$ for any $(\mathbf{N}_1^T, -\vec{s})$-kernel basis $\bar{\mathbf{Q}}_1$ and then apply Lemma 4.2. Note that there exists a polynomial matrix $\bar{\mathbf{Q}}_2$ such that $\bar{\mathbf{Q}}_1\bar{\mathbf{Q}}_2 = \bar{\mathbf{G}}$ for any $(\mathbf{N}^T, -\vec{s})$-kernel basis $\bar{\mathbf{G}}$, as $\bar{\mathbf{G}}$ satisfies $\mathbf{N}_1^T\bar{\mathbf{G}} = 0$ and is therefore generated by the $(\mathbf{N}_1^T, -\vec{s})$-kernel basis $\bar{\mathbf{Q}}_1$. If $\text{cdeg}_{-\vec{s}}\bar{\mathbf{Q}}_1 \not\leq 0$, then Lemma 2.1 forces $\text{cdeg}_{-\vec{s}}\left(\bar{\mathbf{Q}}_1\bar{\mathbf{Q}}_2\right) = \text{cdeg}_{-\vec{s}}\bar{\mathbf{G}} \not\leq 0$,a contradiction since we know from Lemma 4.3 that $\text{cdeg}_{-\vec{s}}\bar{\mathbf{G}} \leq 0$.

As before, to see that a $\left(\mathbf{N}_2^T\mathbf{Q}_1, \vec{b}_2+1, -\vec{s}_2\right)$-basis contains a $(\mathbf{N}_2^T\mathbf{Q}_1, -\vec{s}_2)$-kernel basis whose $-\vec{s}$-column degrees are no more than 0, we can just show $\text{cdeg}_{-\vec{s}_2}\hat{\mathbf{Q}}_2 \leq 0$ for any $(\mathbf{N}_2^T\mathbf{Q}_1, -\vec{s}_2)$-kernel basis $\hat{\mathbf{Q}}_2$ and then apply Lemma 4.2. Since $\text{cdeg}_{\vec{s}}\mathbf{N}_2 = \vec{b}_2$, we have $\text{rdeg}_{-\vec{b}_2}\mathbf{N}_2 \leq -\vec{s}$ or equivalently, $\text{cdeg}_{-\vec{b}_2}\mathbf{N}_2^T \leq -\vec{s}$. Then combining this with $\text{cdeg}_{-\vec{s}}\mathbf{Q}_1 = -\vec{s}_2$ we get $\text{cdeg}_{-\vec{b}_2}\mathbf{N}_2^T\mathbf{Q}_1 \leq -\vec{s}_2$ using Lemma 2.1. Let $\hat{\mathbf{G}} = \mathbf{Q}_1\hat{\mathbf{Q}}_2$, which is a $(\mathbf{N}^T, -\vec{s})$-kernel basis by Theorem 2.10. Note that $\text{cdeg}_{-\vec{s}_2}\hat{\mathbf{Q}}_2 = \text{cdeg}_{-\vec{s}}\mathbf{Q}_1\hat{\mathbf{Q}}_2 = \text{cdeg}_{-\vec{s}}\hat{\mathbf{G}} \leq 0$. □

Algorithm 1 MinimalKernelBasisReversed$(\mathbf{M}, \vec{s}, \xi)$

(Kernel basis computation with reversed degree structure)

Input: $\mathbf{M} \in \mathbb{K}[x]^{k\times n}$ and $\vec{s} \in \mathbb{Z}_{\geq 0}^n$ such that $\sum \text{rdeg}_{\vec{s}}\mathbf{M} \leq \xi$, $\sum \vec{s} \leq \xi$, and any $(\mathbf{M}, -\vec{s})$-kernel basis having row degrees bounded by $\vec{s}$ (equivalently, having $-\vec{s}$-column degrees bounded by 0).

Output: $\mathbf{G} \in \mathbb{K}[x]^{n\times *}$, a $(\mathbf{M}, -\vec{s})$-kernel basis.

1: $\left[\mathbf{M}_1^T, \mathbf{M}_2^T, \cdots, \mathbf{M}_{\lceil \log k\rceil -1}^T, \mathbf{M}_{\lceil \log k\rceil}^T\right] := \mathbf{M}^T$, with $\mathbf{M}_{\lceil \log k\rceil}, \mathbf{M}_{\lceil \log k\rceil -1}, \cdots, \mathbf{M}_2, \mathbf{M}_1$ having $\vec{s}$-row degrees in the range $\left[0, \frac{2\xi}{k}\right], (\frac{2\xi}{k}, \frac{4\xi}{k}], ..., (\frac{\xi}{4}, \frac{\xi}{2}], (\frac{\xi}{2}, \xi]$.
2: **for** i **from** 1 **to** $\lceil \log k\rceil$ **do**
3: $\quad\sigma_i := \left\lceil \frac{\xi}{2^{i-1}}\right\rceil + 1; \vec{\sigma}_i := [\sigma_i, \ldots, \sigma_i]$, number of entries matching the row dimension of $\mathbf{M}_i$;
4: **end for**
5: $\vec{\sigma} := \left[\vec{\sigma}_1, \vec{\sigma}_2, \ldots, \vec{\sigma}_{\lceil \log k\rceil}\right]$;
6: $\hat{\mathbf{N}} := x^{\vec{\sigma}-\vec{b}-1}\mathbf{M}$;
7: $\mathbf{G}_0 := I_n$; $\tilde{\mathbf{G}}_0 := I_n$;
8: **for** i **from** 1 **to** $\lceil \log k\rceil$ **do**
9: $\quad\vec{s}_i := -\text{cdeg}_{-\vec{s}}\mathbf{G}_{i-1}$; (note $\vec{s}_1 = \vec{s}$)
10: $\quad\mathbf{P}_i := \text{UnbalancedFastOrderBasis}\left(\hat{\mathbf{N}}_i\tilde{\mathbf{G}}_{i-1}, \sigma_i, -\vec{s}_i\right)$;
11: $\quad[\mathbf{G}_i, \mathbf{Q}_i] := \mathbf{P}_i$, where $\mathbf{G}_i$ is a $\left(\hat{\mathbf{M}}_i, -\vec{s}_i\right)$-kernel basis;
12: $\quad\tilde{\mathbf{G}}_i := \tilde{\mathbf{G}}_{i-1}\cdot\mathbf{G}_i$;
13: **end for**
14: **return** $\tilde{\mathbf{G}}_i$

4.2 Efficient Computation of Kernel Bases

Now that we can correctly compute a $(\mathbf{N}^T, -\vec{s})$-kernel basis by rows with the help of order basis computation using Lemma 4.4, we need to look at how to do this efficiently. One major difficulty is that the order $\vec{b}+1$, or equivalently, the $\vec{s}$-row degrees of $\mathbf{N}^T$ may be unbalanced and can have degree as large as $\sum \vec{s}$. Note that the existing kernel basis algorithm from [20] handles input matrices with unbalanced column degrees, but not unbalanced row degrees. For example, in the simpler special case of $\vec{s} = [s, \ldots, s]$ having uniformly equal entries, the sum of the row degrees is $O(ns)$, but the sum of column degrees can be $\Theta\left(n^2 s\right)$, which puts an extra factor n to the cost if the algorithm from [20] is used. To overcome this problem with unbalanced $\vec{s}$-row degrees, we separate the rows of $\mathbf{N}^T$ into blocks according to their $\vec{s}$-row degrees, and then work with these blocks one by one successively using Lemma 4.4.

Let k be the column dimension of $\mathbf{N}$ and ξ be an upper bound of $\sum \vec{s}$. Since

$$\sum \text{cdeg}_{\vec{s}}\mathbf{N} = \sum \vec{b} \leq \sum \vec{s} \leq \xi$$

by Theorem 2.9, at most $\frac{k}{c}$ columns of $\mathbf{N}$ have $\vec{s}$-column degrees greater than or equal to $\frac{c\,\xi}{k}$ for any $c \geq 1$. Without loss of generality we can assume that the rows of $\mathbf{N}^T$ are arranged in decreasing $\vec{s}$-row degrees. We divide $\mathbf{N}^T$ into $\lceil \log k\rceil$ row blocks according to the $\vec{s}$-row degrees of its rows,

or equivalently, divide $\mathbf{N}$ into blocks of columns according to the $\vec{s}$-column degrees. Let $\mathbf{N} = \left[\mathbf{N}_1, \mathbf{N}_2, \cdots, \mathbf{N}_{\lceil \log k \rceil - 1}, \mathbf{N}_{\lceil \log k \rceil}\right]$ with $\mathbf{N}_{\lceil \log k \rceil}, \mathbf{N}_{\lceil \log k \rceil - 1}, \ldots, \mathbf{N}_2, \mathbf{N}_1$ having $\vec{s}$-column degrees in the range $[0, 2\xi/k]$, $(2\xi/k, 4\xi/k]$, $(4\xi/k, 8\xi/k]$, ..., $(\xi/4, \xi/2]$, $(\xi/2, \xi]$, respectively. Let $\sigma_i = \left\lceil \xi/2^{i-1} \right\rceil + 1$ and $\vec{\sigma}_i = [\sigma_i, \ldots, \sigma_i]$ with the same dimension as the row dimension of $\mathbf{N}_i$ and $\vec{\sigma} = \left[\vec{\sigma}_{\lceil \log k \rceil}, \vec{\sigma}_{\lceil \log k \rceil - 1}, \ldots, \vec{\sigma}_1\right]$ be the orders in the order basis computation.

To further simplify our task, we also make the order of our problem in each block uniform. Rather than of using $\mathbf{N}^T$ as the input matrix, we instead use

$$\hat{\mathbf{N}} = \begin{bmatrix} \hat{\mathbf{N}}_1 \\ \vdots \\ \hat{\mathbf{N}}_{\lceil \log k \rceil} \end{bmatrix} = x^{\vec{\sigma} - \vec{b} - 1} \begin{bmatrix} \mathbf{N}_1^T \\ \vdots \\ \mathbf{N}_{\lceil \log k \rceil}^T \end{bmatrix} = x^{\vec{\sigma} - \vec{b} - 1} \mathbf{N}^T$$

so that a $\left(\hat{\mathbf{N}}, \vec{\sigma}, -\vec{s}\right)$-order basis is a $\left(\mathbf{N}^T, \vec{b} + 1, -\vec{s}\right)$-order basis.

In order to compute a $\left(\mathbf{N}^T, -\vec{s}\right)$-kernel basis we determine a series of kernel bases via a series of order basis computations as follows:

1. Let $\vec{s}_1 = \vec{s}$. Compute an $\left(\hat{\mathbf{N}}_1, \vec{\sigma}_1, -\vec{s}_1\right)$-order basis $\mathbf{P}_1$ using Algorithm 2 from [19] for order basis computation with unbalanced shift. Note that here the order $\vec{\sigma}_1 = [\sigma_1, \ldots, \sigma_1]$ is uniform, an $\left(\hat{\mathbf{N}}_1, \vec{\sigma}_1, -\vec{s}_1\right)$ order basis is also $\left(\hat{\mathbf{N}}_1, \sigma_1, -\vec{s}_1\right)$-order basis. Partition $\mathbf{P}_1$ as $\mathbf{P}_1 = [\mathbf{G}_1, \mathbf{Q}_1]$, where $\mathbf{G}_1$ is a $\left(\hat{\mathbf{N}}_1, -\vec{s}_1\right)$-kernel basis by Lemma 4.4. Set $\tilde{\mathbf{G}}_1 = \mathbf{G}_1$ and $\vec{s}_2 = -\text{cdeg}_{-\vec{s}} \mathbf{G}_1$.

2. Compute an $\left(\hat{\mathbf{N}}_2 \tilde{\mathbf{G}}_1, \sigma_2, -\vec{s}_2\right)$-order basis $\mathbf{P}_2$ and partition $\mathbf{P}_2 = [\mathbf{G}_2, \mathbf{Q}_2]$ with $\mathbf{G}_2$ a $\left(\hat{\mathbf{N}}_2, -\vec{s}_2\right)$-kernel basis. Set $\vec{s}_3 = -\text{cdeg}_{-\vec{s}_2} \mathbf{G}_2$ and $\tilde{\mathbf{G}}_2 = \tilde{\mathbf{G}}_1 \mathbf{G}_2$.

3. Continuing this process, at each step i we compute a $\left(\hat{\mathbf{N}}_i \tilde{\mathbf{G}}_{i-1}, \sigma_i, -\vec{s}_i\right)$-order basis $\mathbf{P}_i$ and then partition $\mathbf{P}_i = [\mathbf{G}_i, \mathbf{Q}_i]$ with $\mathbf{G}_i$ a $\left(\hat{\mathbf{N}}_i \tilde{\mathbf{G}}_{i-1}, -\vec{s}_i\right)$-kernel basis. Let $\tilde{\mathbf{G}}_i = \prod_{j=1}^{i} \mathbf{G}_i = \tilde{\mathbf{G}}_{i-1} \mathbf{G}_i$.

4. Return $\tilde{\mathbf{G}}_{\lceil \log k \rceil}$, a $\left(\mathbf{N}^T, -\vec{s}\right)$-kernel basis.

This process of computing a $\left(\mathbf{N}^T, -\vec{s}\right)$-kernel basis is formally given in Algorithm 1.

4.3 Cost of Left Kernel Basis Computation

The cost of Algorithm 1 is dominated by the order basis computations and the multiplications $\hat{\mathbf{N}}_i \tilde{\mathbf{G}}_{i-1}$ and $\tilde{\mathbf{G}}_{i-1} \mathbf{G}_i$. Let $s = \xi/n$.

LEMMA 4.5. *An $\left(\hat{\mathbf{N}}_i \tilde{\mathbf{G}}_{i-1}, \sigma_i, -\vec{s}_i\right)$-order basis can be computed with a cost of $O^{\sim}(n^{\omega} s)$.*

PROOF. Note that $\mathbf{N}_i$ has less than 2^i columns. Otherwise, since $\text{cdeg}_{\vec{s}} \mathbf{N}_i > \xi/2^i$, we have $\sum \text{cdeg}_{\vec{s}} \mathbf{N}_i > 2^i \xi / 2^i = \xi$, contradicting with $\sum \text{cdeg}_{\vec{s}} \mathbf{N} = \sum \vec{b} \leq \sum \vec{s} \leq \xi$. It follows that $\hat{\mathbf{N}}_i$, and therefore $\hat{\mathbf{N}}_i \tilde{\mathbf{G}}_{i-1}$, also have less than 2^i rows. We also have $\sigma_i = \left\lceil \xi/2^{i-1} \right\rceil + 1 \in \Theta\left(\xi/2^i\right)$. Therefore, Algorithm 2 from [19] for order basis computation with unbalanced shift can be used with a cost of $O^{\sim}(n^{\omega} s)$. □

LEMMA 4.6. *The multiplications $\hat{\mathbf{N}}_i \tilde{\mathbf{G}}_{i-1}$ can be done with a cost of $O^{\sim}(n^{\omega} s)$.*

PROOF. The dimension of $\hat{\mathbf{N}}_i$ is bounded by $2^i \times n$ and $\sum \text{rdeg}_{\vec{s}} \hat{\mathbf{N}}_i \leq 2^i \cdot \xi / 2^{i-1} \in O(\xi)$. We also have $\text{cdeg}_{-\vec{s}} \tilde{\mathbf{G}}_{i-1} \leq 0$, or equivalently, $\text{rdeg}\, \tilde{\mathbf{G}}_{i-1} \leq \vec{s}$. We can now use Theorem 2.3 to multiply $\tilde{\mathbf{G}}_{i-1}^T$ and $\hat{\mathbf{N}}_i^T$ with a cost of $O^{\sim}(n^{\omega} s)$. □

LEMMA 4.7. *The multiplication $\tilde{\mathbf{G}}_{i-1} \mathbf{G}_i$ can be done with a cost of $O^{\sim}(n^{\omega} s)$.*

PROOF. We know $\text{cdeg}_{-\vec{s}} \tilde{\mathbf{G}}_{i-1} = -\vec{s}_i$, and $\text{cdeg}_{-\vec{s}_i} \mathbf{G}_i = -\vec{s}_{i+1} \leq 0$. In other words, $\text{rdeg}\, \mathbf{G}_i \leq \vec{s}_i$, and $\text{rdeg}_{\vec{s}_i} \tilde{\mathbf{G}}_{i-1} \leq \vec{s}$, hence we can again use Theorem 2.3 to multiply $\mathbf{G}_i^T$ and $\tilde{\mathbf{G}}_{i-1}^T$ with a cost of $O^{\sim}(n^{\omega} s)$. □

LEMMA 4.8. *Given an input matrix $\mathbf{M} \in \mathbb{K}[x]^{k \times n}$, a shift $\vec{s} \in \mathbb{Z}^n$, and an upper bound $\xi \in \mathbb{Z}$ such that*

(i) $\sum \text{rdeg}_{\vec{s}} \mathbf{M} \leq \xi$,

(ii) $\sum \vec{s} \leq \xi$,

(iii) any $(\mathbf{M}, -\vec{s})$-kernel basis having row degrees bounded by $\vec{s}$, or equivalently, $-\vec{s}$-column degrees bounded by 0.

Then Algorithm 1 costs $O^{\sim}(n^{\omega} s)$ field operations to compute a $(\mathbf{M}, -\vec{s})$-kernel basis.

Note that while the upper bound ξ can be simply replaced by $\sum \vec{s}$ in Lemma 4.8 and Algorithm 1 for computing a right factor in this section, keeping it separate makes the algorithm more general and allows it to be reused in the next section.

It may also be informative to note again the correspondence between Lemma 4.8 and Theorem 2.11, on the reversal of the degree structures of the input matrices and the output kernel bases.

THEOREM 4.9. *A right factor $\mathbf{G}$ satisfying $\mathbf{F} = \mathbf{T}\mathbf{G}$ for a column basis $\mathbf{T}$ can be computed with a cost of $O^{\sim}(n^{\omega} s)$.*

5. COMPUTING A COLUMN BASIS

Once a right factor $\mathbf{G}$ of $\mathbf{F}$ has been computed, we are in a position to determine a column basis $\mathbf{T}$ using the equation $\mathbf{F} = \mathbf{T}\mathbf{G}$. In order to do so efficiently, however, the degree of $\mathbf{T}$ cannot be too large. We see that this is the case from the following lemma.

LEMMA 5.1. *Let $\mathbf{F}$ and $\mathbf{G}$ be as before and $\vec{t} = -\text{rdeg}_{-\vec{s}} \mathbf{G}$. Then*

(i) the column degrees of $\mathbf{T}$ are bounded by the corresponding entries of $\vec{t}$;

(ii) if $\vec{t}$ has r entries and $\vec{s}\,'$ is the list of the r largest entries of $\vec{s}$, then $\vec{t} \leq \vec{s}\,'$.

PROOF. Since $\mathbf{G}$ is $-\vec{s}$-row reduced, and $\text{rdeg}_{-\vec{s}} \mathbf{F} \leq 0$, by Lemma 2.1 $\text{rdeg}_{-\vec{t}} \mathbf{T} \leq 0$, or equivalently, $\mathbf{T}$ has column degrees bounded by $\vec{t}$.

Let $\mathbf{G}'$ be the $-\vec{s}$-row Popov form of $\mathbf{G}$ and the square matrix $\mathbf{G}''$ consist of only the columns of $\mathbf{G}'$ that contains pivot entries, and has the rows permuted so the pivots are in the diagonal. Let $\vec{s}\,''$ be the list of the entries in $\vec{s}$ that correspond to the columns of $\mathbf{G}''$ in $\mathbf{G}'$. Note that $\text{rdeg}_{-\vec{s}\,''} \mathbf{G}'' = -\vec{t}\,''$ is just a permutation of $-\vec{t}$ with the

same entries. By the definition of shifted row degree, $-\vec{t}\,''$ is the sum of $-\vec{s}\,''$ and the list of the diagonal pivot degrees, which are nonnegative. Therefore, $-\vec{t}\,'' \geq -\vec{s}\,''$. The result then follows as $\vec{t}$ is a permutation of $\vec{t}\,''$ and $\vec{s}\,'$ consists of the largest entries of $\vec{s}$. □

Having determined a bound on the column degrees of $\mathbf{T}$, we are now ready to compute $\mathbf{T}$. This is done again by computing a kernel basis using an order basis computation as before.

LEMMA 5.2. *Let $\vec{t}^* = [0, \ldots, 0, \vec{t}] \in \mathbb{Z}^{m+r}$. Then any $\left(\left[\mathbf{F}^T, \mathbf{G}^T\right], -\vec{t}^*\right)$-kernel basis has the form $\begin{bmatrix} V \\ \bar{\mathbf{T}} \end{bmatrix}$, where $V \in \mathbb{K}^{m\times m}$ is a unimodular matrix and $\left(\bar{\mathbf{T}}V^{-1}\right)^T$ is a column basis of $\mathbf{F}$.*

PROOF. Note first that the matrix $\begin{bmatrix} -I \\ \mathbf{T}^T \end{bmatrix}$ is a kernel basis of $\left[\mathbf{F}^T, \mathbf{G}^T\right]$ and is therefore unimodularly equivalent to any other kernel basis. Hence any other kernel basis has the form $\begin{bmatrix} -I \\ \mathbf{T}^T \end{bmatrix} U = \begin{bmatrix} V \\ \bar{\mathbf{T}} \end{bmatrix}$, with U and $V = -U$ unimodular. Thus $\mathbf{T} = \left(\bar{\mathbf{T}}V^{-1}\right)^T$. Also note that the $-\vec{t}^*$ minimality forces the unimodular matrix V in any $\left(\left[\mathbf{F}^T, \mathbf{G}^T\right], -\vec{t}^*\right)$-kernel basis to be of degree 0, the same degree as I. □

EXAMPLE 5.3. *Let*

$$\mathbf{F} = \begin{bmatrix} x^2 & x^2 & x+x^2 & 1+x^2 \\ 1+x+x^2 & x^2 & 1+x^2 & 1+x^2 \end{bmatrix},$$

a matrix over $\mathbb{Z}_2[x]$, and

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ x & x^2 & 0 & 1+x^2 \end{bmatrix},$$

a minimal left kernel basis of a right kernel basis of $\mathbf{F}$. In order to compute the column basis $\mathbf{T}$ satisfying $\mathbf{F} = \mathbf{TG}$, first we can determine $\operatorname{cdeg}\mathbf{T} \leq \vec{t} = [2, 0]$ from Lemma 5.1. Then we can compute a $[0, 0, -\vec{t}]$-minimal left kernel basis of $\begin{bmatrix} \mathbf{F} \\ \mathbf{G} \end{bmatrix}$. The matrix

$$[V, \bar{\mathbf{T}}] = \begin{bmatrix} 1 & 0 & x+x^2 & 1 \\ 1 & 1 & 1+x & 0 \end{bmatrix}$$

is such a left kernel basis. A column basis can then be computed as

$$\mathbf{T} = V^{-1}\bar{\mathbf{T}} = \begin{bmatrix} x+x^2 & 1 \\ 1+x^2 & 1 \end{bmatrix}.$$

□

In order to compute a $\left(\left[\mathbf{F}^T, \mathbf{G}^T\right], -\vec{t}^*\right)$-kernel basis, we can again use order basis computation as before, as we again have an order basis that contains a $\left(\left[\mathbf{F}^T, \mathbf{G}^T\right], -\vec{t}^*\right)$-kernel basis.

LEMMA 5.4. *Any $\left(\left[\mathbf{F}^T, \mathbf{G}^T\right], \vec{s}+1, -\vec{t}^*\right)$-order basis contains a $\left(\left[\mathbf{F}^T, \mathbf{G}^T\right], -\vec{t}^*\right)$-kernel basis whose $-\vec{t}^*$-row degrees are bounded by 0.*

PROOF. As before, Lemma 4.2 can be used here. We just need to show that a $\left(\left[\mathbf{F}^T, \mathbf{G}^T\right], -\vec{t}^*\right)$-kernel basis has $-\vec{t}^*$-row degrees no more than 0. This follows from the fact that $\operatorname{rdeg}_{-\vec{t}^*}\begin{bmatrix} I \\ \mathbf{T}^T \end{bmatrix} \leq 0$. □

Algorithm 2 ColumnBasis($\mathbf{F}$)

Input: $\mathbf{F} \in \mathbb{K}[x]^{m\times n}$.
Output: a column basis of $\mathbf{F}$.

1: $\vec{s}$:= $\operatorname{cdeg}\mathbf{F}$;
2: $\mathbf{N}$:= MinimalKernelBasis $(\mathbf{F}, \vec{s})$;
3: $\mathbf{G} := \left(\text{MinimalKernelBasisReversed}(\mathbf{N}^T, \vec{s}, \sum\vec{s})\right)^T$;
4: $\vec{t}^*$:= $[0, \ldots, 0, -\operatorname{rdeg}_{-\vec{s}}\mathbf{G}]$, with rowDimension($\mathbf{G}$) number of 0's ;
5: $\begin{bmatrix} V \\ \bar{\mathbf{T}} \end{bmatrix}$:= MinimalKernelBasisReversed($\left[\mathbf{F}^T, \mathbf{G}^T\right], \vec{t}^*, \sum\vec{s}$) with a square V;
6: $\mathbf{T}$:= $\left(\bar{\mathbf{T}}V^{-1}\right)^T$;
7: **return** $\mathbf{T}$;

In order to compute a $\left(\left[\mathbf{F}^T, \mathbf{G}^T\right], -\vec{t}^*\right)$-kernel basis efficiently, we notice that we have the same type of problem as in Section 4.2 and hence we can again use Algorithm 1.

LEMMA 5.5. *A $\left(\left[\mathbf{F}^T, \mathbf{G}^T\right], -\vec{t}^*\right)$-kernel basis can be computed using Algorithm 1 with a cost of $O^\sim(n^\omega s)$, where $s = \xi/n$ is the average column degree of $\mathbf{F}$ as before.*

PROOF. Just use the algorithm with input $\left(\left[\mathbf{F}^T, \mathbf{G}^T\right], \vec{t}^*, \xi\right)$. We can verify the conditions on the input are satisfied.

- To see that $\sum\operatorname{rdeg}_{\vec{t}^*}\left[\mathbf{F}^T, \mathbf{G}^T\right] \leq \xi$, note that from $\vec{t} = -\operatorname{rdeg}_{-\vec{s}}\mathbf{G}$ and Lemma 2.2 that $\operatorname{cdeg}_{\vec{t}}\mathbf{G} \leq \vec{s}$, or equivalently, $\operatorname{rdeg}_{\vec{t}}\mathbf{G}^T \leq \vec{s}$. Since we also have $\operatorname{rdeg}\mathbf{F}^T \leq \vec{s}$, it follows that $\operatorname{rdeg}_{\vec{t}^*}\left[\mathbf{F}^T, \mathbf{G}^T\right] \leq \vec{s}$.
- The condition $\sum\vec{t}^* \leq \xi$ follows from Lemma 5.1.
- The third condition holds since $\begin{bmatrix} -I \\ \mathbf{T}^T \end{bmatrix}$ is a kernel basis with row degrees bounded by $\vec{t}^*$.

□

With a $\left(\left[\mathbf{F}^T, \mathbf{G}^T\right], -\vec{t}^*\right)$-kernel basis $[V^T, \bar{\mathbf{T}}^T]^T$ computed, a column basis is then given by $\mathbf{T} = \left(\bar{\mathbf{T}}V^{-1}\right)^T$.

The complete algorithm for computing a column basis is then given in Algorithm 2.

THEOREM 5.6. *A column basis $\mathbf{T}$ of $\mathbf{F}$ can be computed with a cost of $O^\sim(n^\omega s)$, where $s = \xi/n$ is the average column degree of $\mathbf{F}$ as before.*

PROOF. The cost is dominated by the cost of the three kernel basis computations in the algorithm. The first one is handled by the algorithm from [20] and Theorem 2.11, while the remaining two are handled by Algorithm 1, Lemma 4.8 and Lemma 5.5. □

6. A SIMPLE IMPROVEMENT

When the input matrix $\mathbf{F}$ has column dimension n much larger than the row dimension m, then we can separate $\mathbf{F} = \left[\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_{n/m}\right]$ into n/m blocks, each with dimension $m \times m$, assuming without loss of generality n is a multiple of m, and the columns are arranged in increasing degrees. We then do a series of column basis computations. First we compute a column basis $\mathbf{T}_1$ of $[\mathbf{F}_1, \mathbf{F}_2]$. Then compute a column basis $\mathbf{T}_2$ of $[\mathbf{T}_1, \mathbf{F}_3]$. Repeating this process, at step i, we compute a column basis $\mathbf{T}_i$ of $[\mathbf{T}_{i-1}, \mathbf{F}_{i+1}]$, until $i = n/m - 1$, when a column basis of $\mathbf{F}$ is computed.

LEMMA 6.1. *Let* $\bar{s}_i = (\sum \operatorname{cdeg} \mathbf{F}_i)/m$. *Then at step* i, *computing a column basis* $\mathbf{T}_i$ *of* $[\mathbf{T}_{i-1}, \mathbf{F}_{i+1}]$ *can be done with a cost of* $O^\sim(m^\omega(\bar{s}_i + \bar{s}_{i+1})/2)$ *field operations.*

PROOF. From Lemma 5.1, the column basis $\mathbf{T}_{i-1}$ of $[\mathbf{F}_1, \ldots, \mathbf{F}_i]$ has column degrees bounded by the largest column degrees of $\mathbf{F}_i$, hence $\sum \operatorname{cdeg} \mathbf{T}_{i-1} \leq \sum \operatorname{cdeg} \mathbf{F}_i$. The lemma then follows by combining this with the result from Theorem 5.6 that a column basis $\mathbf{T}_i$ of $[\mathbf{T}_{i-1}, \mathbf{F}_{i+1}]$ can be computed with a cost of $O^\sim(m^\omega \hat{s}_i)$, where

$$\hat{s}_i = \left(\sum \operatorname{cdeg} \mathbf{T}_{i-1} + \sum \operatorname{cdeg} \mathbf{F}_{i+1}\right)/2m \leq \frac{(\bar{s}_i + \bar{s}_{i+1})}{2}.$$

□

THEOREM 6.2. *If* $s = (\sum \operatorname{cdeg} \mathbf{F})/n$, *then a column basis of* $\mathbf{F}$ *can be computed with a cost of* $O^\sim(m^\omega s)$.

PROOF. Summing up the cost of all the column basis computations,

$$\sum_{i=1}^{n/m-1} O^\sim\left(m^\omega\left(\bar{s}_i + \bar{s}_{i+1}\right)/2\right) \subset O^\sim\left(m^\omega\left(\sum_{i=1}^{n/m} \bar{s}_i\right)\right) = O^\sim\left(nm^{\omega-1}s\right),$$

since $\sum \operatorname{cdeg} \mathbf{F} = \sum_{i=1}^{n/m} (m\bar{s}_i) = ns$. □

REMARK 6.3. *In this section, the computational efficiency is improved by reducing the original problem to about* n/m *subproblems whose column dimensions are close to the row dimension* m. *This is done by successive column basis computations. Note that we can also reduce the column dimension by using successive order basis computations, and only do a column basis computation at the very last step. The computational complexity of using order basis computation to reduce the column dimension would remain the same, but in practice it may be more efficient since order basis computations are simpler.*

7. CONCLUSION

In this paper we have given a fast, deterministic algorithm for the computation of a column basis for $\mathbf{F}$ having complexity $O^\sim(n^\omega s)$ field operations in $\mathbb{K}$ with s an upper bound for the average column degree of $\mathbf{F}$. Our methods rely on a special factorization of $\mathbf{F}$ into a column basis and a kernel basis. These in turn are computed via fast kernel basis and fast order basis algorithm of [20, 19]. When these computations involve the multiplication of polynomial matrices with unbalanced degrees then they use the fast method for such multiplications given in [20].

In a later publication we will show how this column basis algorithm can be used in efficient deterministic computations of matrix determinant, Hermite form, and computations of column reduced form and Popov form for matrices of any rank and any dimension.

8. REFERENCES

[1] B. Beckermann, H. Cheng, and G. Labahn. Fraction-free row reduction of matrices of Ore polynomials. *Journal of Symbolic Computation*, 41(1):513–543, 2006.

[2] B. Beckermann and G. Labahn. A uniform approach for the fast computation of matrix-type Padé approximants. *SIAM Journal on Matrix Analysis and Applications*, 15(3):804–823, 1994.

[3] B. Beckermann and G. Labahn. Recursiveness in matrix rational interpolation problems. *Journal of Computational and Applied Math*, 5-34, 1997.

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

[5] B. Beckermann, G. Labahn, and G. Villard. Shifted normal forms of polynomial matrices. In *Proceedings of ISSAC'99*, pages 189–196, 1999.

[6] B. Beckermann, G. Labahn, and G. Villard. Normal forms for general polynomial matrices. *Journal of Symbolic Computation*, 41(6):708–737, 2006.

[7] Th. G. J. Beelen, G. J. van den Hurk, and C. Praagman. A new method for computing a column reduced polynomial matrix. *System Control Letters*, 10(4):217–224, 1988.

[8] P. Giorgi, C.-P. Jeannerod, and G. Villard. On the complexity of polynomial matrix computations. In *Proceedings of ISSAC'03*, pages 135–142, 2003.

[9] S. Gupta, S. Sarkar, A. Storjohann, and J. Valeriote. Triangular x-basis decompositions and derandomization of linear algebra algorithms over K[x]. *Journal of Symbolic Computation*, 47(4):422–453, 2012.

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

[11] C. Li. Lattice compression of polynomial matrices. Master's thesis, School of Computer Science, University of Waterloo, 2006.

[12] S. Sarkar. Computing Popov forms of polynomial matrices. Master's thesis, University of Waterloo, 2011.

[13] S. Sarkar and A. Storjohann. Normalization of row reduced matrices. In *Proceedings of ISSAC'11*, pages 297–304, 2011.

[14] A. Storjohann and G. Villard. Computing the rank and a small nullspace basis of a polynomial matrix. In *Proceedings of ISSAC'05*, pages 309–316, 2005.

[15] G. Villard. Computing Popov and Hermite forms of polynomial matrices. In *Proceedings of ISSAC'96*, pages 250–258, 1996.

[16] J. von zur Gathen and J Gerhard. *Modern Computer Algebra*. Cambridge University Press, 2003.

[17] W. Zhou. *Fast Order Basis and Kernel Basis Computation and Related Problems*. PhD thesis, University of Waterloo, 2012.

[18] W. Zhou and G. Labahn. Efficient computation of order bases. In *Proceedings of ISSAC'09*, pages 375–382. ACM, 2009.

[19] W. Zhou and G. Labahn. Efficient algorithms for order basis computation. *Journal of Symbolic Computation*, 47:793–819, 2012.

[20] W. Zhou, G. Labahn, and A. Storjohann. Computing minimal nullspace bases. In *Proceedings of ISSAC'12*, pages 375–382. ACM, 2012.