Fast, deterministic computation of the Hermite normal form and determinant of a polynomial matrix

George Labahn, Vincent Neiger, Wei Zhou

Fast, deterministic computation of the Hermite normal form and determinant of a polynomial matrix

George Labahn[1,*], Vincent Neiger[2], Wei Zhou[1]

Abstract

Given a nonsingular $n \times n$ matrix of univariate polynomials over a field $\mathbb{K}$, we give fast and deterministic algorithms to compute its determinant and its Hermite normal form. Our algorithms use $\widetilde{\mathcal{O}}(n^\omega \lceil s \rceil)$ operations in $\mathbb{K}$, where s is bounded from above by both the average of the degrees of the rows and that of the columns of the matrix and ω is the exponent of matrix multiplication. The soft-$\mathcal{O}$ notation indicates that logarithmic factors in the big-$\mathcal{O}$ are omitted while the ceiling function indicates that the cost is $\widetilde{\mathcal{O}}(n^\omega)$ when $s = o(1)$. Our algorithms are based on a fast and deterministic triangularization method for computing the diagonal entries of the Hermite form of a nonsingular matrix.

Keywords: Hermite normal form, determinant, polynomial matrix.

1. Introduction

For a given nonsingular polynomial matrix $\mathbf{A}$ in $\mathbb{K}[x]^{n\times n}$, one can find a unimodular matrix $\mathbf{U} \in \mathbb{K}[x]^{n\times n}$ such that $\mathbf{AU} = \mathbf{H}$ is triangular. Unimodular means that there is a polynomial inverse matrix, or equivalently, the determinant is a nonzero constant from $\mathbb{K}$. Triangularizing a matrix is useful for solving linear systems and computing matrix operations such as determinants or normal forms. In the latter case, the best-known example is the Hermite normal form, first defined by Hermite in 1851 in the context of triangularizing integer matrices [18]. Here,

$$\mathbf{H} = \begin{bmatrix} h_{11} & & & \\ h_{21} & h_{22} & & \\ \vdots & \vdots & \ddots & \\ h_{n1} & \cdots & \cdots & h_{nn} \end{bmatrix}$$

with the added properties that each h_{ii} is monic and $\deg(h_{ij}) < \deg(h_{ii})$ for all $j < i$. Classical variations of this definition include specifying upper rather than lower triangular forms, and specifying row rather than column forms. In the latter case, the unimodular matrix multiplies on the left rather than the right, and the degree of the diagonal entries dominates that of their columns rather than their rows.

*Corresponding author.

Email addresses: glabahn@uwaterloo.ca (George Labahn), vincent.neiger@ens-lyon.fr (Vincent Neiger), w2zhou@uwaterloo.ca (Wei Zhou)

[1]David R. Cheriton School of Computer Science, University of Waterloo, Waterloo ON, Canada N2L 3G1

[2]ENS de Lyon (Laboratoire LIP, CNRS, Inria, UCBL, Université de Lyon), Lyon, France

The goal of this paper is the fast, deterministic computation of the determinant and Hermite normal form of a nonsingular polynomial matrix. The common ingredient in both algorithms is a method for the fast computation of the diagonal entries of a matrix triangularization. The product of these entries gives, at least up to a constant, the determinant while Hermite forms are determined from a given triangularization by reducing the remaining entries modulo the diagonal entries.

In the case of determinant computation, there has been a number of efforts directed to obtaining algorithms whose complexities are given in terms of exponents of matrix multiplication. Interestingly enough, in the case of matrices over a field, Bunch and Hopcroft [9] showed that if there exists an algorithm which multiplies $n \times n$ matrices in $\mathcal{O}(n^\omega)$ field operations for some ω, then there also exists an algorithm for computing the determinant with the same cost bound $\mathcal{O}(n^\omega)$. In the case of an arbitrary commutative ring or of the integers, fast determinant algorithms have been given by Kaltofen [22], Abbott *et al.* [1] and Kaltofen and Villard [23]. We refer the reader to the last named paper and the references therein for more details on efficient determinant computation of such matrices.

In the specific case of the determinant of a matrix of polynomials $\mathbf{A}$ with $\deg(\mathbf{A}) = d$, Storjohann [29] gave a recursive deterministic algorithm making use of fraction-free Gaussian elimination with a cost of $\widetilde{\mathcal{O}}(n^{\omega+1}d)$ operations. A deterministic $\mathcal{O}(n^3d^2)$ algorithm was later given by Mulders and Storjohann [26], modifying their algorithm for weak Popov form computation. Using low rank perturbations, Eberly *et al.* [12] gave a randomized determinant algorithm for integer matrices which can be adapted to be used with polynomial matrices using $\widetilde{\mathcal{O}}(n^{3.5}d)$ field operations. Storjohann [30] later used high order lifting to give a randomized algorithm which computes the determinant using $\widetilde{\mathcal{O}}(n^\omega d)$ field operations. The algorithm of Giorgi *et al.* [13] has a similar cost but only works on a class of generic input matrices, matrices that are well behaved in the computation.

Similarly there has been considerable progress in the efficient computation of the Hermite form of a polynomial matrix. Hafner and McCurley [17] and Iliopoulos [19] give algorithms with a complexity bound of $\widetilde{\mathcal{O}}(n^4d)$ operations from $\mathbb{K}$ where $d = \deg(\mathbf{A})$. They control the size of the matrices encountered during the computation by working modulo the determinant. Using matrix multiplication the algorithms of Hafner and McCurley [17], Storjohann and Labahn [33] and Villard [35] reduce the cost to $\widetilde{\mathcal{O}}(n^{\omega+1}d)$ operations where ω is the exponent of matrix multiplication. The algorithm of Storjohann and Labahn worked with integer matrices but the results directly carry over to polynomial matrices. Mulders and Storjohann [26] then gave an iterative algorithm having complexity $\mathcal{O}(n^3d^2)$, thus reducing the exponent of n but at the cost of increasing the exponent of d.

During the past two decades, there has been a goal to design algorithms that perform various $\mathbb{K}[x]$-linear algebra operations in about the time that it takes to multiply two polynomial matrices having the same dimension and degree as the input matrix, namely at a cost $\widetilde{\mathcal{O}}(n^\omega d)$. *Randomized* algorithms with such a cost already exist for a number of polynomial matrix problems, for example for linear system solving [30], Smith normal form computation [30], row reduction [13] and small nullspace bases computation [34]. In the case of polynomial matrix inversion, the randomized algorithm in [32] costs $\widetilde{\mathcal{O}}(n^3d)$, which is quasi-linear in the number of field elements used to represent the inverse. For Hermite form computation, Gupta and Storjohann [16] gave a randomized algorithm with expected cost $\widetilde{\mathcal{O}}(n^3d)$, later improved to $\widetilde{\mathcal{O}}(n^\omega d)$ in [14]. Their algorithm was the first to be both softly cubic in n and softly linear in d. It is worth mentioning that all the algorithms cited in this paragraph are of the Las Vegas type.

Recently, *deterministic* fast algorithms have been given for linear system solving and row reduc-

tion [15], minimal nullspace bases [41], and matrix inversion [42]. Having a deterministic algorithm has advantages. As a simple but important example, this allows for use over a small finite field $\mathbb{K}$ without the need for resorting to field extensions. The previous fastest Hermite form algorithms [16, 14] do require such field extensions. In this paper, we give deterministic fast algorithms for computing Hermite forms and determinants.

Our approach relies on an efficient method for determining the diagonal elements of a triangularization of the input matrix $\mathbf{A}$. We can do this recursively by determining, for each integer k, a partition

$$\mathbf{A} \cdot \mathbf{U} = \begin{bmatrix} \mathbf{A}_u \\ \mathbf{A}_d \end{bmatrix} \begin{bmatrix} \mathbf{U}_\ell & \mathbf{U}_r \end{bmatrix} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ * & \mathbf{B}_2 \end{bmatrix} = \mathbf{B}$$

where $\mathbf{A}_u$ has k rows, $\mathbf{U}_\ell$ has k columns and $\mathbf{B}_1$ is of size $k \times k$. The subscripts for $\mathbf{A}$ and $\mathbf{U}$ are meant to denote up, down, left and right. As $\mathbf{A}$ is nonsingular, $\mathbf{A}_u$ has full rank and hence one has that $\mathbf{U}_r$ is a basis of the kernel of $\mathbf{A}_u$. Furthermore the matrix $\mathbf{B}_1$ is nonsingular and is therefore a column basis of $\mathbf{A}_u$.

However the recursion described above requires additional properties if it is to be efficient for our applications. In the case of determinants, $\mathbf{A} \cdot \mathbf{U}$ being lower triangular implies that we need both the product of the diagonals and also the determinant of the unimodular multiplier. For the case of Hermite form computation a sensible approach would be to first determine a triangular form of $\mathbf{A}$ and then reduce the lower triangular elements using the diagonal entries with unimodular operations. In both applications it appears that we would need to know $\mathbf{U} = \mathbf{A}^{-1}\mathbf{H}$. However the degrees in such a unimodular multiplier can be too large for efficient computation. Indeed there are examples where the sum of the degrees in $\mathbf{U}$ is $\Theta(n^3 d)$ (see Section 3), in which case computing $\mathbf{U}$ is beyond our target cost $\widetilde{\mathcal{O}}(n^\omega d)$.

In order to achieve the desired efficiency, our triangularization computations need to be done without actually determining the entire unimodular matrix $\mathbf{U}$. We accomplish this by making use of shifted minimal kernel bases and column bases of polynomial matrices, whose computations can be done efficiently using algorithms from [41] and [39]. Shifts are weightings of column degrees which basically help us to control the computations using column degrees rather than the degree of the polynomial matrix. Using the degree becomes an issue for efficiency when the degrees in the input matrix vary considerably from column to column. We remark that shifted minimal kernel bases and column bases, used in the context of fast block elimination, have also been used for deterministic algorithms for inversion [42] and unimodular completion [40] of polynomial matrices.

Fast algorithms for computing shifted minimal kernel bases [41] and column bases [39] imply that we can deterministically find the diagonals in $\widetilde{\mathcal{O}}(n^\omega \lceil s \rceil)$ field operations, where s is the average of the column degrees of $\mathbf{A}$. We recall that the ceiling function indicates that for matrices with very low average column degree $s \in o(1)$, this cost is still $\widetilde{\mathcal{O}}(n^\omega)$. By modifying this algorithm slightly we can also compute the determinant of the unimodular multiplier, giving our first contribution. In the next theorem, $D(\mathbf{A})$ is the so-called *generic determinant bound* as defined in [15] (see also Section 2.3). It has the important property that $D(\mathbf{A})/n$ is bounded from above by both the average of the degrees of the columns of $\mathbf{A}$ and that of its rows.

Theorem 1.1. *Let $\mathbf{A}$ be a nonsingular matrix in $\mathbb{K}[x]^{n \times n}$. There is a deterministic algorithm which computes the determinant of $\mathbf{A}$ using $\widetilde{\mathcal{O}}(n^\omega \lceil D(\mathbf{A})/n \rceil) \subseteq \widetilde{\mathcal{O}}(n^\omega \lceil s \rceil)$ operations in $\mathbb{K}$, with s being the minimum of the average of the degrees of the columns of $\mathbf{A}$ and that of its rows.*

Applying our fast diagonal entry algorithm for Hermite form computation has more technical

challenges. The difficulty comes from the unpredictability of the diagonal degrees of $\mathbf{H}$, which coincide with its row degrees. Indeed, we know that the sum of the diagonal degrees in $\mathbf{H}$ is $\deg(\det(\mathbf{A})) \leq nd$, and so the sum of the degrees in $\mathbf{H}$ is $\mathcal{O}(n^2 d)$. Still, the best known *a priori* bound for the degree of the i-th diagonal entry is $(n-i+1)d$ and hence the sum of these bounds is $\mathcal{O}(n^2 d)$, a factor of n larger than the actual sum. Determining the diagonal entries gives us the row degrees of $\mathbf{H}$ and thus solves this issue. Still, it remains a second major task: that of computing the remaining entries of $\mathbf{H}$.

The randomized algorithm of Gupta and Storjohann [16, 14] solves the Hermite form problem using two steps, which both make use of the Smith normal form $\mathbf{S}$ of $\mathbf{A}$ and partial information on a left multiplier $\mathbf{V}$ for this Smith form. The matrices $\mathbf{S}$ and $\mathbf{V}$ can be computed with a Las Vegas randomized algorithm using an expected number of $\widetilde{\mathcal{O}}(n^\omega d)$ field operations [16, 14], relying in particular on high-order lifting [30, Section 17]. The first step of their algorithm consists of computing the diagonal entries of $\mathbf{H}$ by triangularization of a $2n \times 2n$ matrix involving $\mathbf{S}$ and $\mathbf{V}$, a computation done in $\widetilde{\mathcal{O}}(n^\omega d)$ operations [14]. The second step sets up a system of linear modular equations which admits $\mathbf{A}$ as a basis of solutions: the matrix of the system is $\mathbf{V}$ and the moduli are the diagonal entries of $\mathbf{S}$. The degrees of the diagonal entries obtained in the first step are then used to find $\mathbf{H}$ as another basis of solutions of this system, computed in $\widetilde{\mathcal{O}}(n^\omega d)$ [16] using in particular fast minimal approximant basis and partial linearization techniques [31, 38].

The algorithm presented here for Hermite forms follows a two-step process similar to the algorithm of Gupta and Storjohann, but it avoids using the Smith form of $\mathbf{A}$, whose deterministic computation in $\widetilde{\mathcal{O}}(n^\omega d)$ still remains an open problem. Instead, as explained above, we compute the diagonal entries of $\mathbf{H}$ deterministically via equation (1) using $\widetilde{\mathcal{O}}(n^\omega \lceil s \rceil)$ field operations, where s is the average of the column degrees of $\mathbf{A}$. As for the second step, using the knowledge of the diagonal degrees of $\mathbf{H}$ combined with partial linearization techniques from [15, Section 6], we show that $\mathbf{H}$ can then be computed via a single call to fast deterministic column reduction [15] using $\widetilde{\mathcal{O}}(n^\omega d)$ field operations. This new problem reduction illustrates the fact that knowing in advance the degree shape of reduced or normal forms makes their computation much easier, something already observed and exploited in [16, 36, 20].

This approach results in a deterministic $\widetilde{\mathcal{O}}(n^\omega d)$ algorithm for Hermite form computation, which is satisfactory for matrices $\mathbf{A}$ that have most entries of similar degree $d = \deg(\mathbf{A})$. However, inspired from other contexts such as approximant and kernel basis computations [31, 38, 20, 41] as well as polynomial matrix inversion [42] and the determinant algorithm in this paper, one may hope for algorithms that are even faster than $\widetilde{\mathcal{O}}(n^\omega d)$ when the degrees in $\mathbf{A}$ are non-uniform, for example, if all high-degree entries are located in a few rows and columns of $\mathbf{A}$. In the present paper we use ideas in [15] to reduce the non-uniformity of the degrees in $\mathbf{A}$ in the context of Hermite form computation, thus obtaining Theorem 1.2.

Theorem 1.2. *Let $\mathbf{A}$ be a nonsingular matrix in $\mathbb{K}[x]^{n\times n}$. There is a deterministic algorithm which computes the Hermite form of $\mathbf{A}$ using $\widetilde{\mathcal{O}}(n^\omega \lceil D(\mathbf{A})/n \rceil) \subseteq \widetilde{\mathcal{O}}(n^\omega \lceil s \rceil)$ operations in $\mathbb{K}$, with s being the minimum of the average of the degrees of the columns of $\mathbf{A}$ and that of its rows.*

The remainder of this paper is organized as follows. In Section 2 we give preliminary information on shifted degrees as well as kernel and column bases of polynomial matrices. We also recall why it is interesting to have cost bounds involving the generic determinant bound rather than the degree of the matrix; see in particular Remark 2.6. Section 3 contains the fast algorithm for finding the diagonal entries of a triangular form. This is followed in Section 4 by our algorithm for finding the

determinant. The reduction of degrees of off diagonal entries in the Hermite form is then given in Section 5. It computes the remaining entries by relying in particular on fast deterministic column reduction. In Section 6 we then give the details about how to use partial linearization to decrease the non-uniformity of the degrees in the input matrix for Hermite form computation. The paper ends with a conclusion and topics for future research.

2. Preliminaries

In this section we first give the basic notations for *column degrees* and *shifted degrees* of vectors and matrices of polynomials. We then present the building blocks used in our algorithms, namely the concepts of *kernel basis* and *column basis* for a matrix of polynomials. Finally, we explain our interest in having cost bounds involving the so-called *generic determinant bound*.

2.1. Shifted Degrees

Our methods make use of the concept of *shifted* degrees of polynomial matrices [7], basically shifting the importance of the degrees in some of the rows of a basis. For a column vector $\mathbf{p} = [p_1, \ldots, p_n]^\mathsf{T}$ of univariate polynomials over a field $\mathbb{K}$, its column degree, denoted by $\mathrm{cdeg}(\mathbf{p})$, is the maximum of the degrees of the entries of $\mathbf{p}$, that is,

$$\mathrm{cdeg}(\mathbf{p}) = \max_{1 \le i \le n} \deg(p_i).$$

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

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

where

$$\mathbf{x}^{\vec{s}} = \mathrm{Diag}\left(x^{s_1}, x^{s_2}, \ldots, x^{s_n}\right).$$

For a matrix $\mathbf{P}$, we use $\mathrm{cdeg}(\mathbf{P})$ and $\mathrm{cdeg}_{\vec{s}}(\mathbf{P})$ to denote respectively the list of its column degrees and the list of its shifted $\vec{s}$-column degrees. For the *uniform shift* $\vec{s} = (0, \ldots, 0)$, the shifted column degree specializes to the standard column degree. Similarly, $\mathrm{cdeg}_{-\vec{s}}(\mathbf{P}) \le 0$ is equivalent to $\deg(p_{ij}) \le s_i$ for all i and j, that is, $\vec{s}$ bounds the row degrees of $\mathbf{P}$.

The shifted row degree of a row vector $\mathbf{q} = [q_1, \ldots, q_n]$ is defined similarly as

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

Shifted degrees have been used previously in polynomial matrix computations and in generalizations of some matrix normal forms [8]. The shifted column degree is equivalent to the notion of *defect* commonly used in the rational approximation literature.

Along with shifted degrees we also make use of the notion of a polynomial matrix being column reduced. A full-rank polynomial matrix $\mathbf{A} = [a_{ij}]_{i,j}$ is column reduced if its leading column coefficient matrix, that is the matrix

$$\mathrm{lm}(\mathbf{A}) = [\mathrm{coeff}(a_{ij}, x, d_j)]_{1 \le i,j \le n}, \text{ with } (d_1, \ldots, d_n) = \mathrm{cdeg}(\mathbf{A}),$$

has full rank. Then, the polynomial matrix $\mathbf{A}$ is $\vec{s}$-column reduced if $\mathbf{x}^{\vec{s}}\,\mathbf{A}$ is column reduced. The concept of $\mathbf{A}$ being shifted row reduced is similar.

The usefulness of the shifted degrees can be seen from their applications in polynomial matrix computation problems such as Hermite-Padé and M-Padé approximations [4, 2, 5, 38], minimal kernel bases [41], and shifted column reduction [8, 27].

An essential fact needed in this paper, also based on the use of shifted degrees, is the efficient multiplication of matrices with unbalanced degrees [41, Theorem 3.7].

Theorem 2.1. *Let $\mathbf{A} \in \mathbb{K}[x]^{m\times n}$ with $m \le n$, $\vec{s} \in \mathbb{N}^n$ 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 $k \in \mathcal{O}(m)$ and the sum θ of its $\vec{s}$-column degrees satisfying $\theta \in \mathcal{O}(\xi)$. Then we can multiply $\mathbf{A}$ and $\mathbf{B}$ with a cost of $\widetilde{\mathcal{O}}(n^2 m^{\omega-2}\lceil s\rceil) \subseteq \widetilde{\mathcal{O}}(n^\omega \lceil s\rceil)$, where $s = \xi/n$ is the average of the entries of $\vec{s}$.*

2.2. Shifted Kernel and Column Bases

The kernel of $\mathbf{A} \in \mathbb{K}[x]^{m\times n}$ is the $\mathbb{K}[x]$-module $\{\mathbf{p} \in \mathbb{K}[x]^{n\times 1} \mid \mathbf{Ap} = 0\}$. Such a module is free and of rank $k \le n$ [11, Chapter 12, Theorem 4]; any of its bases is called a kernel basis of $\mathbf{A}$. In other words:

Definition 2.2. *Given $\mathbf{A} \in \mathbb{K}[x]^{m\times n}$, a polynomial matrix $\mathbf{N} \in \mathbb{K}[x]^{n\times k}$ is a (right) kernel basis of $\mathbf{A}$ if the following properties hold:*

1. *$\mathbf{N}$ has full rank,*
2. *$\mathbf{N}$ satisfies $\mathbf{A}\cdot\mathbf{N} = 0$,*
3. *Any $\mathbf{q} \in \mathbb{K}[x]^{n\times 1}$ satisfying $\mathbf{Aq} = 0$ can be written as a linear combination of the columns of $\mathbf{N}$, that is, there exists $\mathbf{p} \in \mathbb{K}[x]^{k\times 1}$ such that $\mathbf{q} = \mathbf{Np}$.*

It is easy to show that any pair of kernel bases $\mathbf{N}$ and $\mathbf{M}$ of $\mathbf{A}$ are unimodularly equivalent. An $\vec{s}$-minimal kernel basis of $\mathbf{A}$ is a kernel basis that is $\vec{s}$-column reduced.

Definition 2.3. *Given $\mathbf{A} \in \mathbb{K}[x]^{m\times n}$, a matrix $\mathbf{N} \in \mathbb{K}[x]^{n\times k}$ is an $\vec{s}$-minimal (right) kernel basis of $\mathbf{A}$ if $\mathbf{N}$ is a kernel basis of $\mathbf{A}$ and $\mathbf{N}$ is $\vec{s}$-column reduced.*

A column basis of $\mathbf{A}$ is a basis of the $\mathbb{K}[x]$-module $\{\mathbf{Ap},\ \ \mathbf{p} \in \mathbb{K}[x]^{n\times 1}\}$, which is free of rank $r \le n$. Such a basis can be represented as a full rank matrix $\mathbf{M} \in \mathbb{K}[x]^{m\times r}$ whose columns are the basis elements. A column basis is not unique and indeed any column basis right multiplied by a unimodular matrix gives another column basis.

Example 2.4. Let

$$\mathbf{A} = \begin{bmatrix} 6x+1 & 2x^3+x^2+6x+1 & 3 \\ 4x^5+5x^4+4x^2+x & 6x^5+5x^4+2x^3+4 & x^4+5x^3+6x^2+5x \end{bmatrix}$$

be a 2×3 matrix over $\mathbb{Z}_7[x]$ having column degree $\vec{s} = (5,5,4)$. Then a column basis $\mathbf{B}$, and a kernel basis $\mathbf{N}$, of $\mathbf{A}$ are given by

$$\mathbf{B} = \begin{bmatrix} 5x+5 & 1 \\ 3 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{N} = \begin{bmatrix} 6x^6+4x^5+5x^4+3x^3+4x^2+1 \\ 4x^4+5x^3+x^2+6x \\ 4x^7+4x^6+4x^5+4x^3+5x^2+3x+2 \end{bmatrix}.$$

For example, if $\mathbf{b}_1$ and $\mathbf{b}_2$ denote the columns of $\mathbf{B}$ then the third column of $\mathbf{A}$, denoted by $\mathbf{a}_3$, is given by

$$\mathbf{a}_3 = (4x^3 + 3x^2 + 6x + 5)\,\mathbf{b}_1 \;+\; (x^4 + 4x^2 + x + 6)\,\mathbf{b}_2.$$

Here $\operatorname{cdeg}_{\vec{s}}(\mathbf{N}) = (11)$. In addition, the shifted leading coefficient matrix

$$\operatorname{lm}_{\vec{s}}(\mathbf{N}) = \begin{bmatrix} 6 \\ 0 \\ 4 \end{bmatrix}$$

has full rank, and hence we have that $\mathbf{N}$ is an $\vec{s}$-minimal kernel basis of $\mathbf{A}$. ◇

Fast algorithms for kernel basis computation and column basis computation are given in [41] and in [39], respectively. In both cases they make use of fast methods for order bases (often also referred to as minimal approximant bases) [5, 13, 37, 38]. In what follows, we write $|\vec{s}|$ for the sum of the entries of a tuple $\vec{s} \in \mathbb{N}^n$ with nonnegative entries.

Theorem 2.5. *Let $\mathbf{A} \in \mathbb{K}[x]^{m\times n}$ with $m \leq n$ and $m \in \Theta(n)$, and let $\vec{s} \in \mathbb{N}^n$ be such that $\operatorname{cdeg}(\mathbf{A}) \leq \vec{s}$ componentwise. Then, there exist deterministic algorithms which compute*

(i) an $\vec{s}$-minimal kernel basis of $\mathbf{A}$ using $\widetilde{\mathcal{O}}(n^\omega \lceil s \rceil)$ field operations,

(ii) a column basis of $\mathbf{A}$ using $\widetilde{\mathcal{O}}(n^\omega \lceil s \rceil)$ field operations,

where $s = |\vec{s}|/n$ is the average column degree of $\mathbf{A}$.

2.3. The generic determinant degree bound

For a nonsingular $n \times n$ matrix $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$, the degree of the determinant of $\mathbf{A}$ provides a good measure of the size of the output $\mathbf{H}$ in the case of Hermite form computation. Indeed, if we denote by $\vec{\delta} = (\delta_1, \ldots, \delta_n)$ the degrees of the diagonal entries of $\mathbf{H}$, then we have $\deg(\det(\mathbf{A})) = \delta_1 + \cdots + \delta_n$. Since the diagonal entries are those of largest degree in their respective rows, we directly obtain that $\mathbf{H}$ can be represented using $n^2 + n|\vec{\delta}| = n^2 + n\deg(\det(\mathbf{A}))$ field elements.

The size of the input $\mathbf{A}$ can be measured by several quantities, which differ in how precisely they account for the distribution of the degrees in $\mathbf{A}$. It is interesting to relate these quantities to the degree of the determinant of $\mathbf{A}$, since the latter measures the size of the output $\mathbf{H}$. A first, coarse bound is given by the maximum degree of the entries of the matrix: $\mathbf{A}$ can be represented by $n^2 + n^2\deg(\mathbf{A})$ field elements. On the other hand, by definition of the determinant we have that $\det(\mathbf{A})$ has degree at most $n\deg(\mathbf{A})$. A second, finer bound can be obtained using the *average* of the row degrees and of the column degrees: the size of $\mathbf{A}$ in terms of field elements is at most $n^2 + n\min(|\operatorname{rdeg}(\mathbf{A})|, |\operatorname{cdeg}(\mathbf{A})|)$. Again we have the related bound

$$\deg(\det(\mathbf{A})) \leq \min(|\operatorname{rdeg}(\mathbf{A})|, |\operatorname{cdeg}(\mathbf{A})|).$$

An even finer bound on the size of $\mathbf{A}$ is given by the *generic determinant bound*, introduced in [15, Section 6]. For $\mathbf{A} = [a_{ij}] \in \mathbb{K}[x]^{n\times n}$, this is defined as

$$D(\mathbf{A}) = \max_{\pi\in S_n} \sum_{1\leq i\leq n} \overline{\deg}(a_{i,\pi_i}) \tag{1}$$

where S_n is the set of permutations of $\{1, \ldots, n\}$, and where

$$\overline{\deg}(p) = \begin{cases} 0 & \text{if } p = 0 \\ \deg(p) & \text{if } p \neq 0 \end{cases}.$$

By definition, we have the inequalities

$$\deg(\det(\mathbf{A})) \leq D(\mathbf{A}) \leq \min(|\text{rdeg}(\mathbf{A})|, |\text{cdeg}(\mathbf{A})|) \leq n \deg(\mathbf{A}),$$

and it is easily checked that $\mathbf{A}$ can be represented using $n^2 + 2nD(\mathbf{A})$ field elements.

Thus in Hermite form computation both the input and the output have average degree in $\mathcal{O}(D(\mathbf{A})/n)$ and can be represented using $\mathcal{O}(n^2\lceil D(\mathbf{A})/n \rceil)$ field elements. Furthermore $D(\mathbf{A})$ gives a more precise account of the degrees in $\mathbf{A}$ than the average row and column degrees, and an algorithm with cost bound $\widetilde{\mathcal{O}}(n^\omega \lceil D(\mathbf{A})/n \rceil)$ is always faster, sometimes significantly, than an algorithm with cost bound $\widetilde{\mathcal{O}}(n^\omega \lceil s \rceil)$ where s is the average column degree or the average row degree, let alone $s = \deg(\mathbf{A})$.

Remark 2.6. Let us justify why this can sometimes be *significantly* faster. We have seen that $D(\mathbf{A})/n$ is bounded from above by both the average column degree and the average row degree of $\mathbf{A}$. It turns out that, in some important cases $D(\mathbf{A})/n$ may be substantially smaller than these averages. For example, consider $\mathbf{A}$ with one row and one column of uniformly large degree d and all other entries of degree 0:

$$\mathbf{A} = \begin{bmatrix} [d] & [d] & \cdots & [d] \\ [d] & [0] & \cdots & [0] \\ \vdots & \vdots & \ddots & \vdots \\ [d] & [0] & \cdots & [0] \end{bmatrix} \in \mathbb{K}[x]^{n \times n}.$$

Here, the average row degree and the average column degree are both exactly d while the generic determinant bound is d as well. Thus, here $D(\mathbf{A})/n = d/n$ is much smaller than $d = \deg(\mathbf{A}) = \min(|\text{rdeg}(\mathbf{A})|/n, |\text{cdeg}(\mathbf{A})|/n)$. For similar examples, we refer the reader to [15, Example 4] and [42, equation (8)]. ◇

3. Determining the diagonal entries of a triangular form

In this section we show how to determine the diagonal entries of a triangular form of a non-singular matrix $\mathbf{A} \in \mathbb{K}[x]^{n \times n}$ with $\mathbf{A}$ having column degrees $\vec{s}$. Our algorithm makes use of fast kernel and column bases computations.

As mentioned in the introduction, we consider unimodularly transforming $\mathbf{A}$ to

$$\mathbf{AU} = \mathbf{B} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ * & \mathbf{B}_2 \end{bmatrix} \tag{2}$$

which eliminates a top right block and gives two square diagonal blocks $\mathbf{B}_1$ and $\mathbf{B}_2$ in $\mathbf{B}$. After this block triangularization step, the matrix is now closer to being in triangular form. Applying this procedure recursively to $\mathbf{B}_1$ and $\mathbf{B}_2$, until the matrices reach dimension 1, gives the diagonal entries of a triangular form of $\mathbf{A}$. These entries are unique up to multiplication by a nonzero constant from $\mathbb{K}$, and in particular making them monic yields the diagonal entries of the Hermite form of $\mathbf{A}$.

In this procedure, a major problem is that the degrees in the unimodular multiplier $\mathbf{U}$ can be too large for efficient computation. For example, the matrix

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ -x^d & 1 & 0 & \cdots & 0 \\ 0 & -x^d & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -x^d & 1 \end{bmatrix} \in \mathbb{K}[x]^{n \times n}$$

of degree $d > 0$ is unimodular and hence its Hermite form is the identity. However the corresponding unimodular multiplier is

$$\mathbf{U} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ x^d & 1 & 0 & \cdots & 0 \\ x^{2d} & x^d & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ x^{(n-1)d} & \cdots & x^{2d} & x^d & 1 \end{bmatrix},$$

with the sum of the degrees in $\mathbf{U}$ being in $\Theta(n^3 d)$, beyond our target cost $\mathcal{O}(n^\omega d)$.

3.1. Fast block elimination

Our approach is to make use of fast kernel and column basis methods to efficiently compute the diagonal blocks $\mathbf{B}_1$ and $\mathbf{B}_2$ while at the same time avoiding the computation of all of $\mathbf{U}$.

Partition $\mathbf{A} = \begin{bmatrix} \mathbf{A}_u \\ \mathbf{A}_d \end{bmatrix}$, with $\mathbf{A}_u$ and $\mathbf{A}_d$ consisting of the upper $\lceil n/2 \rceil$ and lower $\lfloor n/2 \rfloor$ rows of $\mathbf{A}$, respectively. Then both upper and lower parts have full-rank since $\mathbf{A}$ is assumed to be nonsingular. By partitioning $\mathbf{U} = \begin{bmatrix} \mathbf{U}_\ell & \mathbf{U}_r \end{bmatrix}$, where the column dimension of $\mathbf{U}_\ell$ matches the row dimension of $\mathbf{A}_u$, then $\mathbf{A} \cdot \mathbf{U} = \mathbf{B}$ becomes

$$\begin{bmatrix} \mathbf{A}_u \\ \mathbf{A}_d \end{bmatrix} \begin{bmatrix} \mathbf{U}_\ell & \mathbf{U}_r \end{bmatrix} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ * & \mathbf{B}_2 \end{bmatrix}.$$

Notice that the matrix $\mathbf{B}_1$ is nonsingular and is therefore a column basis of $\mathbf{A}_u$. As such this can be efficiently computed as mentioned in Theorem 2.5. In order to compute $\mathbf{B}_2 = \mathbf{A}_d \mathbf{U}_r$, notice that the matrix $\mathbf{U}_r$ is a right kernel basis of $\mathbf{A}_u$, which makes the top right block of $\mathbf{B}$ zero.

The following lemma states that the kernel basis $\mathbf{U}_r$ can be replaced by any other kernel basis of $\mathbf{A}_u$ thus giving another unimodular matrix that also works.

Lemma 3.1. *Partition* $\mathbf{A} = \begin{bmatrix} \mathbf{A}_u \\ \mathbf{A}_d \end{bmatrix}$ *and suppose* $\mathbf{B}_1$ *is a column basis of* $\mathbf{A}_u$ *and* $\mathbf{N}$ *a kernel basis of* $\mathbf{A}_u$*. Then there is a unimodular matrix* $\mathbf{U} = \begin{bmatrix} * & \mathbf{N} \end{bmatrix}$ *such that*

$$\mathbf{AU} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ * & \mathbf{B}_2 \end{bmatrix},$$

where $\mathbf{B}_2 = \mathbf{A}_d \mathbf{N}$*. If* $\mathbf{A}$ *is square and nonsingular, then* $\mathbf{B}_1$ *and* $\mathbf{B}_2$ *are also square and nonsingular.*

Proof. This follows from [39, Lemma 3.1]. □

Algorithm 1 HermiteDiagonal($\mathbf{A}$)

Input: $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$ nonsingular.
Output: $\mathbf{d} \in \mathbb{K}[x]^n$ the list of diagonal entries of the Hermite normal form of $\mathbf{A}$.

1: **if** $n = 1$ **then**
2: write $\mathbf{A} = \lambda\mathbf{d}$ with $\lambda \in \mathbb{K}$ and $\mathbf{d} \in \mathbb{K}[x]$ monic;
3: **return** $\mathbf{d}$;
4: **end if**
5: Partition $\mathbf{A} := \begin{bmatrix}\mathbf{A}_u \\ \mathbf{A}_d\end{bmatrix}$, where $\mathbf{A}_u$ consists of the top $\lceil n/2 \rceil$ rows of $\mathbf{A}$;
6: $\mathbf{B}_1 := \text{ColumnBasis}(\mathbf{A}_u)$;
7: $\mathbf{N} := \text{MinimalKernelBasis}(\mathbf{A}_u, \text{cdeg}(\mathbf{A}))$;
8: $\mathbf{B}_2 := \mathbf{A}_d\mathbf{N}$;
9: $\mathbf{d}_1 := \text{HermiteDiagonal}(\mathbf{B}_1)$;
10: $\mathbf{d}_2 := \text{HermiteDiagonal}(\mathbf{B}_2)$;
11: **return** $[\mathbf{d}_1, \mathbf{d}_2]$;

Note that we do not compute the blocks represented by the symbol $*$. Thus Lemma 3.1 allows us to determine $\mathbf{B}_1$ and $\mathbf{B}_2$ independently without computing the unimodular matrix. This procedure for computing the diagonal entries is presented in Algorithm 1. Formally the cost of this algorithm is given in Proposition 3.3.

3.2. Computational cost and example

Before giving a cost bound for our algorithm, let us observe its correctness on an example.

Example 3.2. Let

$$\mathbf{A} = \begin{bmatrix} 6x+1 & 2x^3+x^2+6x+1 & 3 \\ 4x^5+5x^4+4x^2+x & 6x^5+5x^4+2x^3+4 & x^4+5x^3+6x^2+5x \\ 2 & 2x^5+5x^4+5x^3+6x^2 & 6 \end{bmatrix},$$

working over $\mathbb{Z}_7[x]$. Considering the matrix $\mathbf{A}_u$ formed by the top two rows of $\mathbf{A}$, then a column basis $\mathbf{B}_1$ and kernel basis $\mathbf{N}$ of $\mathbf{A}_u$ were given in Example 2.4. If $\mathbf{A}_d$ denotes the bottom row of $\mathbf{A}$, then this gives diagonal blocks

$$\mathbf{B}_1 = \begin{bmatrix} 5x+5 & 1 \\ 3 & 1 \end{bmatrix}$$

and

$$\mathbf{B}_2 = \mathbf{A}_d\mathbf{N} = \begin{bmatrix} x^9+2x^8+x^7+4x^6+6x^5+4x^4+3x^3+3x^2+4x \end{bmatrix}.$$

Recursively computing with $\mathbf{B}_1$, we obtain a column basis and kernel basis of the top row $\mathbf{B}_{1,u}$ of $\mathbf{B}_1$, as

$$\tilde{\mathbf{B}}_1 = \begin{bmatrix} 1 \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{N}} = \begin{bmatrix} 1 \\ 2x+2 \end{bmatrix}.$$

If $\mathbf{B}_{1,d}$ denote the bottom row of $\mathbf{B}_1$, we get $\tilde{\mathbf{B}}_2 = \mathbf{B}_{1,d}\tilde{\mathbf{N}} = \begin{bmatrix} 2x+5 \end{bmatrix}$, which gives the second diagonal block from $\mathbf{B}_1$. Thus we have the diagonal entries of a triangular form of $\mathbf{B}_1$. On the other hand, since $\mathbf{B}_2$ is already a 1×1 matrix we do not need to do any extra work. As a result

we have that $\mathbf{A}$ is unimodularly equivalent to

$$\begin{bmatrix} 1 & & \\ * & 2x+5 & \\ * & * & x^9+2x^8+x^7+4x^6+6x^5+4x^4+3x^3+3x^2+4x \end{bmatrix},$$

giving, up to making them monic, the diagonal entries of the Hermite form of $\mathbf{A}$. ◇

Proposition 3.3. *Algorithm 1 costs $\widetilde{\mathcal{O}}(n^\omega\lceil s\rceil)$ field operations to compute the diagonal entries of the Hermite normal form of a nonsingular matrix $\mathbf{A}\in\mathbb{K}[x]^{n\times n}$, where $s=|\mathrm{cdeg}(\mathbf{A})|/n$ is the average column degree of $\mathbf{A}$.*

Proof. The three main operations are computing a column basis of $\mathbf{A}_u$, computing a kernel basis $\mathbf{N}$ of $\mathbf{A}_u$, and multiplying the matrices $\mathbf{A}_d\mathbf{N}$. Let $\vec{s}$ denote the column degree of $\mathbf{A}$ and set $\xi=|\vec{s}|$, an integer used to measure size for our problem.

For the column basis computation, by Theorem 2.5 (see also [39, Theorem 5.6]) we know that a column basis $\mathbf{B}_1$ of $\mathbf{A}_u$ can be computed with a cost of $\widetilde{\mathcal{O}}(n^\omega\lceil s\rceil)$, where $s=\xi/n$. Furthermore, the sum of the column degrees of the computed $\mathbf{B}_1$ is bounded by the sum of the column degrees of $\mathbf{A}_u$ (see [39], in particular the proof of Lemma 5.5 therein). Thus, since $\mathrm{cdeg}(\mathbf{A}_u)\le\vec{s}$ componentwise, the sum of the column degrees of $\mathbf{B}_1$ is at most ξ.

Similarly, according to Theorem 2.5 (see also [41, Theorem 4.1]), computing an $\vec{s}$-minimal kernel basis $\mathbf{N}$ of $\mathbf{A}_u$ costs $\widetilde{\mathcal{O}}(n^\omega\lceil s\rceil)$ operations, and the sum of the $\vec{s}$-column degrees of the output kernel basis $\mathbf{N}$ is bounded by ξ [41, Theorem 3.4].

For the matrix multiplication $\mathbf{A}_d\mathbf{N}$, we have that the sum of the column degrees of $\mathbf{A}_d$ and the sum of the $\vec{s}$-column degrees of $\mathbf{N}$ are both bounded by ξ. Therefore Theorem 2.1 applies and the multiplication can be done with a cost of $\widetilde{\mathcal{O}}(n^\omega\lceil s\rceil)$. Furthermore, since the entries of $\vec{s}$ bounds the corresponding column degrees of $\mathbf{A}_d$, according to [41, Lemma 3.1], we have that the column degrees of $\mathbf{B}_2=\mathbf{A}_d\mathbf{N}$ are bounded by the $\vec{s}$-column degrees of $\mathbf{N}$. In particular, the sum of the column degrees of $\mathbf{B}_2$ is at most ξ.

If we let the cost of Algorithm 1 be $g(n)$ for an input matrix of dimension n then

$$g(n) \quad\in\quad \widetilde{\mathcal{O}}(n^\omega\lceil s\rceil)+g(\lceil n/2\rceil)+g(\lfloor n/2\rfloor).$$

As $s=\xi/n$ depends on n we use $\widetilde{\mathcal{O}}(n^\omega\lceil s\rceil)=\widetilde{\mathcal{O}}(n^\omega(s+1))=\widetilde{\mathcal{O}}(n^{\omega-1}\xi+n^\omega)$ with ξ not depending on n. Then we solve the recurrence relation as

$$\begin{aligned} g(n) &\in \widetilde{\mathcal{O}}(n^{\omega-1}\xi+n^\omega)+g(\lceil n/2\rceil)+g(\lfloor n/2\rfloor)\\ &\subseteq \widetilde{\mathcal{O}}(n^{\omega-1}\xi+n^\omega)+2g(\lceil n/2\rceil)\\ &\subseteq \widetilde{\mathcal{O}}(n^{\omega-1}\xi+n^\omega)=\widetilde{\mathcal{O}}(n^\omega\lceil s\rceil). \end{aligned}$$

□

In this cost bound, we do not detail the logarithmic factors because it is not clear to us for the moment how many logarithmic factors arise from the calls to the kernel basis and column basis algorithms of [41, 39], where they are not reported. Yet, from the recurrence relation above, it can be observed that no extra logarithmic factor will be introduced if $\omega>2$, while an extra factor logarithmic in n will be introduced if $\omega=2$.

4. Efficient Determinant Computation

In this section, we show how to recursively and efficiently compute the determinant of a nonsingular matrix $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$ having column degrees $\vec{s}$. Our algorithm follows a strategy similar to the recursive block triangularization in Section 3, making use of fast kernel basis and column basis computation.

Indeed, after unimodularly transforming $\mathbf{A}$ to

$$\mathbf{AU} = \mathbf{B} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ * & \mathbf{B}_2 \end{bmatrix}$$

as in equation (2), the determinant of $\mathbf{A}$ can be computed as

$$\det(\mathbf{A}) = \frac{\det(\mathbf{B})}{\det(\mathbf{U})} = \frac{\det(\mathbf{B}_1)\det(\mathbf{B}_2)}{\det(\mathbf{U})}, \tag{3}$$

which requires us to first compute $\det(\mathbf{B}_1)$, $\det(\mathbf{B}_2)$, and $\det(\mathbf{U})$. The same procedure can then be applied to compute the determinant of $\mathbf{B}_1$ and the determinant of $\mathbf{B}_2$. However, as $\mathbf{U}$ is unimodular we will handle its determinant differently. This can be repeated recursively until the dimension becomes 1.

One major obstacle for efficiency of this approach is that we do want to compute the scalar $\det(\mathbf{U})$, and as noted in Section 3, the degrees of the unimodular matrix $\mathbf{U}$ can be too large for efficient computation. To sidestep this issue, we will show that $\det(\mathbf{U})$ can be computed with only partial knowledge of the matrix $\mathbf{U}$. Combining this with the method of Section 3 to compute the matrices $\mathbf{B}_1$ and $\mathbf{B}_2$ without computing all of $\mathbf{B}$ and $\mathbf{U}$, we obtain an efficient recursive algorithm.

Remark 4.1. In some cases, the computation of the determinant is easily done from the diagonal entries of a triangular form. Indeed, let $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$ be nonsingular and assume that we have computed the diagonal entries $h_{11}, \ldots, h_{nn}$ of its Hermite form. Then, $\det(\mathbf{A}) = \lambda h_{11}\cdots h_{nn}$ for some nonzero constant $\lambda \in \mathbb{K}$. If the constant coefficient of $h_{11}\cdots h_{nn}$ is nonzero, we can retrieve λ by computing the constant coefficient of $\det(\mathbf{A})$, which is found by $\mathbb{K}$-linear algebra using $\mathcal{O}(n^\omega)$ operations since $\det(\mathbf{A})(0) = \det(\mathbf{A}(0))$. More generally, if we know $\alpha \in \mathbb{K}$ such that $h_{11}(\alpha)\cdots h_{nn}(\alpha) \neq 0$, then we can deduce $\det(\mathbf{A})$ efficiently. Yet, this does not lead to a fast deterministic algorithm in general since it may happen that $\det(\mathbf{A})(\alpha) = 0$ for all field elements α, or that finding α with $h_{11}(\alpha)\cdots h_{nn}(\alpha) \neq 0$ is a difficult task. ◇

We now focus on computing the determinant of $\mathbf{U}$, or equivalently, the determinant of $\mathbf{V} = \mathbf{U}^{-1}$. The column basis computation from [39] for computing the $m \times m$ diagonal block $\mathbf{B}_1$ also gives $\mathbf{U}_r$, the matrix consisting of the right $(n-m)$ columns of $\mathbf{U}$, which is a right kernel basis of $\mathbf{A}_u$. In fact, this column basis computation also gives a right factor multiplied with the column basis $\mathbf{B}_1$ to give $\mathbf{A}_u$. The following lemma shows that this right factor coincides with the matrix $\mathbf{V}_u$ consisting of the top m rows of $\mathbf{V}$. The column basis computation therefore gives both $\mathbf{U}_r$ and $\mathbf{V}_u$ with no additional work.

Lemma 4.2. *Let m be the dimension of $\mathbf{B}_1$. The matrix $\mathbf{V}_u \in \mathbb{K}[x]^{m\times n}$ satisfies $\mathbf{B}_1\mathbf{V}_u = \mathbf{A}_u$ if and only if $\mathbf{V}_u$ is the submatrix of $\mathbf{V} = \mathbf{U}^{-1}$ formed by its top m rows.*

Proof. The proof follows directly from

$$\mathbf{BV} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ * & \mathbf{B}_2 \end{bmatrix} \begin{bmatrix} \mathbf{V}_u \\ \mathbf{V}_d \end{bmatrix} = \begin{bmatrix} \mathbf{A}_u \\ \mathbf{A}_d \end{bmatrix} = \mathbf{A} \qquad \square.$$

While the determinant of $\mathbf{V}$ or the determinant of $\mathbf{U}$ is needed to compute the determinant of $\mathbf{A}$, a major problem is that we do not know $\mathbf{U}_\ell$ or $\mathbf{V}_d$, which may not be efficiently computed due to their possibly large degrees. This means we need to compute the determinant of $\mathbf{V}$ or $\mathbf{U}$ without knowing the complete matrix $\mathbf{V}$ or $\mathbf{U}$. The following lemma shows how this can be done using just $\mathbf{U}_r$ and $\mathbf{V}_u$, which are obtained from the computation of the column basis $\mathbf{B}_1$.

Lemma 4.3. *Let* $\mathbf{U} = \begin{bmatrix} \mathbf{U}_\ell & \mathbf{U}_r \end{bmatrix}$ *and* $\mathbf{A}$ *satisfy, as before,*

$$\mathbf{AU} = \begin{bmatrix} \mathbf{A}_u \\ \mathbf{A}_d \end{bmatrix} \begin{bmatrix} \mathbf{U}_\ell & \mathbf{U}_r \end{bmatrix} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ * & \mathbf{B}_2 \end{bmatrix} = \mathbf{B},$$

where the row dimension of $\mathbf{A}_u$*, the column dimension of* $\mathbf{U}_\ell$*, and the dimension of* $\mathbf{B}_1$ *are* m*. Let* $\mathbf{V} = \begin{bmatrix} \mathbf{V}_u \\ \mathbf{V}_d \end{bmatrix}$ *be the inverse of* $\mathbf{U}$ *with* m *rows in* $\mathbf{V}_u$ *and* $\mathbf{U}_\ell^* \in \mathbb{K}[x]^{n \times m}$ *be a matrix such that* $\mathbf{U}^* = \begin{bmatrix} \mathbf{U}_\ell^* & \mathbf{U}_r \end{bmatrix}$ *is unimodular. Then* $\mathbf{V}_u\mathbf{U}_\ell^*$ *is unimodular and*

$$\det(\mathbf{A}) = \frac{\det(\mathbf{B}) \det(\mathbf{V}_u\mathbf{U}_\ell^*)}{\det(\mathbf{U}^*)}.$$

Proof. Since $\det(\mathbf{A}) = \det(\mathbf{B}) \det(\mathbf{V})$, we just need to show that $\det(\mathbf{V}) = \det(\mathbf{V}_u\mathbf{U}_\ell^*)/\det(\mathbf{U}^*)$. This follows from

$$\begin{aligned} \det(\mathbf{V}) \det(\mathbf{U}^*) &= \det(\mathbf{V}\mathbf{U}^*) \\ &= \det\left(\begin{bmatrix} \mathbf{V}_u \\ \mathbf{V}_d \end{bmatrix} \begin{bmatrix} \mathbf{U}_\ell^* & \mathbf{U}_r \end{bmatrix} \right) \\ &= \det\left(\begin{bmatrix} \mathbf{V}_u\mathbf{U}_\ell^* & \mathbf{0} \\ * & \mathbf{I} \end{bmatrix} \right) \\ &= \det(\mathbf{V}_u\mathbf{U}_\ell^*). \end{aligned}$$

In particular $\det(\mathbf{V}_u\mathbf{U}_\ell^*)$ is a nonzero constant and thus $\mathbf{V}_u\mathbf{U}_\ell^*$ is unimodular. $\square$

Lemma 4.3 shows that the determinant of $\mathbf{V}$ can be computed using $\mathbf{V}_u$, $\mathbf{U}_r$, and a unimodular completion $\mathbf{U}^*$ of $\mathbf{U}_r$. In fact, this can be made more efficient still by noticing that since we are looking for a constant determinant, the higher degree parts of the matrices do not affect the computation. Indeed, if $\mathbf{U} \in \mathbb{K}[x]^{n \times n}$ is unimodular, then one has

$$\det(\mathbf{U}) = \det(\mathbf{U} \bmod x) = \det(\mathbf{U}(0)) \tag{4}$$

since

$$\det(\mathbf{U} \bmod x) = \det(\mathbf{U}(0)) = \det(\mathbf{U})(0) = \det(\mathbf{U}) \bmod x = \det(\mathbf{U}).$$

Equation (4) allows us to use just the degree zero coefficient matrices in the computation. Hence Lemma 4.3 can be improved as follows.

Lemma 4.4. *Let* $\mathbf{A}$, $\mathbf{U} = \begin{bmatrix}\mathbf{U}_\ell & \mathbf{U}_r\end{bmatrix}$, *and* $\mathbf{V} = \begin{bmatrix}\mathbf{V}_u \\ \mathbf{V}_d\end{bmatrix}$ *be as before. Let* $U_r = \mathbf{U}_r \bmod x$ *and* $V_u = \mathbf{V}_u \bmod x$ *be the constant matrices of* $\mathbf{U}_r$ *and* $\mathbf{V}_u$, *respectively. Let* $U_\ell^* \in \mathbb{K}^{n\times m}$ *be a matrix such that* $U^* = \begin{bmatrix}U_\ell^* & U_r\end{bmatrix}$ *is nonsingular. Then*

$$\det(\mathbf{A}) = \frac{\det(\mathbf{B})\det(V_u U_\ell^*)}{\det(U^*)}.$$

Proof. Suppose $\mathbf{U}_\ell^* \in \mathbb{K}[x]^{n\times m}$ is such that $U_\ell^* = \mathbf{U}_\ell^* \bmod x$ and $\mathbf{U}^* = \begin{bmatrix}\mathbf{U}_\ell^* & \mathbf{U}_r\end{bmatrix}$ is unimodular. Using Lemma 4.3 and equation (4), we have that $\mathbf{V}_u\mathbf{U}_\ell^*$ is unimodular with $V_u U_\ell^* = \mathbf{V}_u\mathbf{U}_\ell^* \bmod x$ and thus

$$\det(\mathbf{A}) = \det(\mathbf{B})\det(\mathbf{V}_u\mathbf{U}_\ell^*)/\det(\mathbf{U}^*) = \det(\mathbf{B})\det(V_u U_\ell^*)/\det(U^*).$$

Let us now show how to construct such a matrix $\mathbf{U}_\ell^*$. Let $\mathbf{W}_\ell^* \in \mathbb{K}[x]^{n\times m}$ be any matrix such that $\mathbf{W}^* = \begin{bmatrix}\mathbf{W}_\ell^* & \mathbf{U}_r\end{bmatrix}$ is unimodular and let W_ℓ^* denote its constant term $W_\ell^* = \mathbf{W}_\ell^* \bmod x$. It is easily checked that

$$\begin{bmatrix}W_\ell^* & U_r\end{bmatrix}^{-1}\begin{bmatrix}U_\ell^* & U_r\end{bmatrix} = \begin{bmatrix}T_u & 0\\ T_d & I\end{bmatrix}$$

for some nonsingular $T_u \in \mathbb{K}^{m\times m}$ and some $T_d \in \mathbb{K}^{n-m\times m}$. Define the matrix $\mathbf{U}_\ell^* = \mathbf{W}_\ell^*\begin{bmatrix}T_u\\T_d\end{bmatrix}$ in $\mathbb{K}[x]^{n\times m}$. On the one hand, we have that the matrix $\mathbf{U}^* = \begin{bmatrix}\mathbf{U}_\ell^* & \mathbf{U}_r\end{bmatrix} = \mathbf{W}^*\begin{bmatrix}T_u & 0\\ T_d & I\end{bmatrix}$ is unimodular. On the other hand, by construction we have that $\mathbf{U}_\ell^* \bmod x = W_\ell^*\begin{bmatrix}T_u\\T_d\end{bmatrix} = U_\ell^*$. □

Thus Lemma 4.4 requires us to compute $U_\ell^* \in \mathbb{K}^{n\times m}$ a matrix such that $U^* = \begin{bmatrix}U_\ell^* & U_r\end{bmatrix}$ is nonsingular. This can be obtained from the nonsingular matrix that transforms V_u to its reduced column echelon form computed using the Gauss Jordan transform algorithm from [29] with a cost of $\mathcal{O}(nm^{\omega-1})$ field operations.

We now have all the ingredients needed for computing the determinant of $\mathbf{A}$. A recursive algorithm is given in Algorithm 2, which computes the determinant of $\mathbf{A}$ as the product of the determinant of $\mathbf{V}$ and the determinant of $\mathbf{B}$. The determinant of $\mathbf{B}$ is computed by recursively computing the determinants of its diagonal blocks $\mathbf{B}_1$ and $\mathbf{B}_2$.

Proposition 4.5. *Algorithm 2 costs* $\widetilde{\mathcal{O}}(n^\omega\lceil s\rceil)$ *field operations to compute the determinant of a nonsingular matrix* $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$, *where* s *is the average column degree of* $\mathbf{A}$.

Proof. From Lemma 3.1 and Proposition 3.3 the computation of the two diagonal blocks $\mathbf{B}_1$ and $\mathbf{B}_2$ costs $\widetilde{\mathcal{O}}(n^\omega\lceil s\rceil)$ field operations. As mentioned above, computing U_l^* at Step 6 of the algorithm costs $\mathcal{O}(n^\omega)$ operations. Step 7 involves only constant matrices so that d_V can be computed $\mathcal{O}(n^\omega)$. Finally, $\det(\mathbf{B}_1)$ and $\det(\mathbf{B}_2)$ are computed recursively and multiplied. Since these are two univariate polynomials of degree at most $\deg(\det(\mathbf{A})) \le \xi = ns$, their product $\mathbf{d_B}$ is obtained in $\widetilde{\mathcal{O}}(\xi) \subset \widetilde{\mathcal{O}}(n^\omega\lceil s\rceil)$ operations.

Therefore, the recurrence relation for the cost of the Algorithm 2 is the same as that in the proof of Proposition 3.3, and the total cost is $\widetilde{\mathcal{O}}(n^\omega\lceil s\rceil)$. □

Proposition 4.5 can be further improved using the following result from [15, Corollary 3].

Algorithm 2 determinant($\mathbf{A}$)

Input: $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$, nonsingular.
Output: the determinant of $\mathbf{A}$.

1: **if** $n = 1$ **then**
2: **return** $\mathbf{A}$;
3: **end if**
4: $\begin{bmatrix}\mathbf{A}_u \\ \mathbf{A}_d\end{bmatrix} := \mathbf{A}$, with $\mathbf{A}_u$ consisting of the top $\lceil n/2 \rceil$ rows of $\mathbf{A}$;
5: $\mathbf{B}_1, \mathbf{U}_r, \mathbf{V}_u := \text{ColumnBasis}(\mathbf{A}_u)$;
Note: Here ColumnBasis() also returns the kernel basis $\mathbf{U}_r$ and the right factor $\mathbf{V}_u$ such that $\mathbf{A}_u = \mathbf{B}_1\mathbf{V}_u$.
6: $\mathbf{B}_2 := \mathbf{A}_d\mathbf{U}_r$;
7: $U_r := \mathbf{U}_r \bmod x$; $V_u := \mathbf{V}_u \bmod x$;
8: Compute a matrix $U_\ell^* \in \mathbb{K}^{n\times\lceil n/2\rceil}$ such that $U^* = \begin{bmatrix}U_\ell^* & U_r\end{bmatrix}$ is nonsingular;
9: $d_V := \det(V_u U_\ell^*)/\det(U^*)$ (element of $\mathbb{K}$);
10: $\mathbf{d_B} := \text{determinant}(\mathbf{B}_1)\,\text{determinant}(\mathbf{B}_2)$;
11: **return** $d_V\mathbf{d}_B$;

Proposition 4.6. *Let $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$ be nonsingular. Using no operation in $\mathbb{K}$, one can build a matrix $\hat{\mathbf{A}} \in \mathbb{K}[x]^{\hat{n}\times\hat{n}}$ such that*

(i) $n \le \hat{n} < 3n$ and $\deg(\hat{\mathbf{A}}) \le \lceil D(\mathbf{A})/n \rceil$,

(ii) the determinant of $\mathbf{A}$ is equal to the determinant of $\hat{\mathbf{A}}$.

This reduction, combined with our result in Proposition 4.5, proves Theorem 1.1.

Example 4.7. In order to observe the correctness of the algorithm, let

$$\mathbf{A} = \begin{bmatrix} -x+2 & -2x-3 & 3x^3+x^2 & -x+2 & -3x^5-x^4 \\ -x & -2 & 3x^3 & -x & -3x^5 \\ -2 & x+3 & 2 & -2 & -2x^2 \\ 0 & 1 & -3x^2-2 & -2x^2-1 & x^4+x^2 \\ 0 & 2 & 3 & -3x^2 & -2x^4-3x^2+3 \end{bmatrix}$$

working over $\mathbb{Z}_7[x]$. If $\mathbf{A}_u$ denotes the top three rows of $\mathbf{A}$, then we have a column basis

$$\mathbf{B}_1 = \begin{bmatrix} -x+2 & -2x-3 & 3x^3+x^2 \\ -x & -2 & 3x^3 \\ -2 & x+3 & 2 \end{bmatrix}$$

and a minimal kernel basis

$$\mathbf{U}_r = \begin{bmatrix} 3 & 0 \\ 0 & 0 \\ 0 & x^2 \\ -3 & 0 \\ 0 & 1 \end{bmatrix}$$

for $\mathbf{A}_u$. The second block diagonal is then given by

$$\mathbf{A}_d\mathbf{U}_r = \begin{bmatrix} x^2-3 & -2x^4-x^2 \\ -2x^2 & -2x^4+3 \end{bmatrix}.$$

The computation of the column basis $\mathbf{B}_1$ also gives the right factor

$$\mathbf{V}_u = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -x^2 \end{bmatrix}$$

and so the constant term matrices are then

$$U_r = \begin{bmatrix} 3 & 0 \\ 0 & 0 \\ 0 & 0 \\ -3 & 0 \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad V_u = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

with Gaussian-Jordan elimination used to find a nonsingular completion of U_r as

$$U_\ell^* = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

The determinant of $\mathbf{U}$ is then computed as

$$d_V = \frac{\det(V_u U_\ell^*)}{\det(U^*)} = -\frac{1}{3} = 2$$

where we recall that $U^* = [U_\ell^* \;\; U_r]$. The determinants of $\mathbf{B}_1$ and $\mathbf{B}_2$ are then computed recursively. In the case of $\mathbf{B}_1$ a minimal kernel basis and column basis are given by

$$\mathbf{U}_{r,1} = \begin{bmatrix} 3x^2 \\ 0 \\ 1 \end{bmatrix}, \quad \mathbf{B}_{1,1} = \begin{bmatrix} -x+2 & 0 \\ -x & 2x-2 \end{bmatrix}, \quad \text{and} \;\; \mathbf{V}_{u,1} = \begin{bmatrix} 1 & 2 & -3x^2 \\ 0 & 1 & 0 \end{bmatrix}.$$

This gives the remaining diagonal block as $\mathbf{B}_{1,2} = \begin{bmatrix} x^2+2 \end{bmatrix}$. The corresponding constant term matrices $U_{r,1}$ and $V_{u,1}$ and nonsingular completion $U_{\ell,1}^*$ are then given by

$$U_{r,1} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad V_{u,1} = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad \text{and} \;\; U_{\ell,1}^* = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix},$$

which gives $d_{V_1} = 1$. Hence $\det(\mathbf{B}_1) = (-x+2)(2x-2)(x^2+2)$. A similar argument gives

$\det(\mathbf{B}_2) = (x^2-3)(x^4+3)$ and hence

$$\det(\mathbf{A}) = d_V \det(\mathbf{B}_1)\det(\mathbf{B}_2) = 3x^{10} - 2x^9 + 3x^8 + 2x^7 - x^6 - x^5 + x^4 - x^3 - 2x^2 + x - 3. \quad \diamond$$

5. Fast computation of the Hermite form

In Section 3, we have shown how to efficiently determine the diagonal entries of the Hermite normal form of a nonsingular input matrix $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$. One then still needs to determine the remaining entries for the complete Hermite form $\mathbf{H}$ of $\mathbf{A}$.

Here, we observe that knowing the diagonal degrees of $\mathbf{H}$ allows us to use partial linearization techniques [15, Section 6] to reduce to the case of computing a column reduced form of $\mathbf{A}$ for an almost uniform shift. Along with the algorithm in Section 3, this gives an algorithm to compute the Hermite form of $\mathbf{A}$ in $\widetilde{\mathcal{O}}(n^\omega \deg(\mathbf{A}))$ field operations using fast deterministic column reduction [15].

5.1. Hermite form via shifted column reduction

It is known that the Hermite form $\mathbf{H}$ of $\mathbf{A}$ is a *shifted* reduced form of $\mathbf{A}$ for a whole range of shifts. Without further information on the degrees in $\mathbf{H}$, one appropriate shift is

$$\vec{h} \;=\; (n(n-1)d,\;\; n(n-2)d,\;\; \ldots,\;\; nd,\;\; 0) \tag{5}$$

where $d = \deg(\mathbf{A})$ (cf. [8, Lemma 2.6]). Note that this shift has a large amplitude, namely $\max(\vec{h}) - \min(\vec{h}) \in \Theta(n^2 d)$. Unfortunately we are not aware of a deterministic shifted reduction algorithm that would compute an $\vec{h}$-reduced form of $\mathbf{A}$ in $\widetilde{\mathcal{O}}(n^\omega d)$ field operations.

Now, let us consider the degrees $\vec{\delta} = (\delta_1, \ldots, \delta_n)$ of the diagonal entries of $\mathbf{H}$. Then we have that $\mathbf{H}$ is a $-\vec{\delta}$-column reduced form of $\mathbf{A}$ and, in addition, that $\mathbf{H}$ can be easily recovered from any $-\vec{\delta}$-column reduced form of $\mathbf{A}$. More precisely, suppose that we know $\vec{\delta}$, for example thanks to the algorithm in Section 3. Then, we claim that $\mathbf{H}$ can be computed as follows, where $\vec{\mu} = (\max(\vec{\delta}), \ldots, \max(\vec{\delta})) \in \mathbb{N}^n$:

$$\mathbf{x}^{\vec{\mu}-\vec{\delta}}\mathbf{A} \xrightarrow{\text{reduction}} \mathbf{x}^{\vec{\mu}-\vec{\delta}}\mathbf{R} \xrightarrow{\text{normalization}} \mathbf{H} = \mathbf{R}\cdot \mathrm{lm}_{-\vec{\delta}}(\mathbf{R})^{-1}$$

where $\mathbf{R}$ is any $-\vec{\delta}$-column reduced form of $\mathbf{A}$. To show this, we will rely on the following consequence of [28, Lemma 17].

Lemma 5.1. *Let $\mathbf{A}$ and $\mathbf{B}$ be column reduced matrices in $\mathbb{K}[x]^{n\times n}$ with uniform column degree $(d, \ldots, d)$, for some $d \in \mathbb{N}$. If $\mathbf{A}$ and $\mathbf{B}$ are right-unimodularly equivalent then*

$$\mathbf{A}\cdot \mathrm{lm}(\mathbf{A})^{-1} = \mathbf{B}\cdot \mathrm{lm}(\mathbf{B})^{-1}.$$

Proof. The matrix $\mathbf{A}$ is column reduced with uniform column degree $(d, \ldots, d)$. As such $\mathbf{A}\cdot\mathrm{lm}(\mathbf{A})^{-1}$ is its Popov form according to [28, Lemma 17] (i.e. its leading coefficient matrix is the identity). Similarly, $\mathbf{B}\cdot\mathrm{lm}(\mathbf{B})^{-1}$ is the Popov form of $\mathbf{B}$ in this case. We recall that the Popov form is a canonical form under right-unimodular equivalence for nonsingular matrices in $\mathbb{K}[x]^{n\times n}$; for a general definition we refer the reader to [21]. Thus, since $\mathbf{A}$ and $\mathbf{B}$ are right-unimodularly equivalent, the uniqueness of the Popov form implies $\mathbf{A}\cdot\mathrm{lm}(\mathbf{A})^{-1} = \mathbf{B}\cdot\mathrm{lm}(\mathbf{B})^{-1}$. □

As we often wish to apply Lemma 5.1 with shifts we also include the following.

Lemma 5.2. *Let $\vec{s} \in \mathbb{Z}^n$ be a shift, and let $\mathbf{A}$ and $\mathbf{B}$ be $\vec{s}$-column reduced matrices in $\mathbb{K}[x]^{n\times n}$ with uniform $\vec{s}$-column degree $(d, \ldots, d)$, for some $d \in \mathbb{Z}$. If $\mathbf{A}$ and $\mathbf{B}$ are right-unimodularly equivalent then*

$$\mathbf{A} \cdot \mathrm{lm}_{\vec{s}}(\mathbf{A})^{-1} = \mathbf{B} \cdot \mathrm{lm}_{\vec{s}}(\mathbf{B})^{-1}.$$

Proof. We simply replace $\mathbf{A}$ and $\mathbf{B}$ by $\mathbf{x}^{\vec{s}}\,\mathbf{A}$ and $\mathbf{x}^{\vec{s}}\,\mathbf{B}$ in the previous proof. □

In addition, since the Hermite form of $\mathbf{A}$ is the shifted Popov form of $\mathbf{A}$ for the shift $\vec{h}$ in equation (5), we can state the following specific case of [20, Lemma 4.1].

Corollary 5.3. *Let $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$ be nonsingular and $\vec{\delta} \in \mathbb{N}^n$ denote the degrees of the diagonal entries of the Hermite form $\mathbf{H}$ of $\mathbf{A}$. If $\mathbf{R}$ is a $-\vec{\delta}$-column reduced form of $\mathbf{A}$, then $\mathbf{R}$ has $-\vec{\delta}$-column degree $\mathrm{cdeg}_{-\vec{\delta}}(\mathbf{R}) = \vec{0}$, row degree $\mathrm{rdeg}(\mathbf{R}) = \vec{\delta}$, and $\mathbf{H} = \mathbf{R} \cdot \mathrm{lm}_{-\vec{\delta}}(\mathbf{R})^{-1}$.*

Proof. Note that $\mathrm{lm}_{-\vec{\delta}}(\mathbf{H})$ is the identity matrix, so that $\mathbf{H}$ is a $-\vec{\delta}$-reduced form of $\mathbf{A}$. Furthermore, $\mathbf{H}$ has $-\vec{\delta}$-column degree $(0, \ldots, 0)$ which implies that $\mathrm{cdeg}_{-\vec{\delta}}(\mathbf{R}) = \vec{0}$ and thus $\mathrm{rdeg}(\mathbf{R}) \leq \vec{\delta}$ componentwise. By Lemma 5.2 we obtain $\mathbf{H} = \mathbf{R} \cdot \mathrm{lm}_{-\vec{\delta}}(\mathbf{R})^{-1}$. In addition, we must have $\mathrm{rdeg}(\mathbf{R}) = \vec{\delta}$, since otherwise $\mathrm{lm}_{-\vec{\delta}}(\mathbf{R})$ would have a zero row. □

Thus we can start with the matrix $\mathbf{x}^{\vec{\mu}-\vec{\delta}}\,\mathbf{A}$, column reduce this matrix and then normalize it to get our normal form. However $\mathbf{x}^{\vec{\mu}-\vec{\delta}}\,\mathbf{A}$ may have some entries of large degree. Indeed, $\max(\vec{\delta})$ may be as large as $\deg(\det(\mathbf{A}))$ while having $\min(\vec{\delta}) = 0$, in which case the degree of $\mathbf{x}^{\vec{\mu}-\vec{\delta}}\,\mathbf{A}$ is at least $\deg(\det(\mathbf{A}))$. For efficient deterministic shifted column reduction we would need the degree of $\mathbf{x}^{\vec{\mu}-\vec{\delta}}\,\mathbf{A}$ to be in $\mathcal{O}(\deg(\mathbf{A}))$.

5.2. Reducing the amplitude of $\vec{\delta}$ using partial linearization

In the strategy presented in the previous subsection, the main obstacle to obtaining an efficient algorithm is that the diagonal degrees of $\mathbf{H}$ might have a large amplitude. In this subsection, we will show how *partial linearization* techniques allow us to build a matrix $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ such that $\mathbf{H}$ can be obtained from a $-\vec{d}$-reduced form of $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ for a shift $\vec{d}$ that has a small amplitude.

A key fact is that the average of the degrees $\vec{\delta}$ is controlled. Namely, denoting by δ the average of $\vec{\delta}$, we have that $\delta \leq \deg(\mathbf{A})$. Indeed, the product of the diagonal entries of $\mathbf{H}$ is $\det(\mathbf{H})$ which, up to a constant multiplier, is the same as $\det(\mathbf{A})$ and thus the degree of this product is

$$n\delta = \delta_1 + \cdots + \delta_n = \deg(\det(\mathbf{A})) \leq n \deg(\mathbf{A}).$$

In order to reduce the amplitude of $\vec{\delta}$, one can split the entries that are larger than δ into several entries each at most δ. From this we obtain another tuple $\vec{d} = (d_1, \ldots, d_{\widetilde{n}})$ with $\max(\vec{d}) - \min(\vec{d}) \leq \delta \leq \deg(\mathbf{A})$ and having length $\widetilde{n}$ less than $2n$.

Most importantly for our purpose, there is a corresponding transformation of matrices which behaves well with regards to shifted reduction. Namely, this transformation is a type of *row partial linearization* [15, Section 6]. Let us consider the case of the Hermite form $\mathbf{H}$ of $\mathbf{A}$. For each i, we consider the row i of $\mathbf{H}$. If its degree δ_i is larger than δ then the row is expanded into α_i rows of degree at most δ. This yields a $\widetilde{n} \times n$ matrix $\widetilde{\mathbf{H}}$ of degree at most δ. Furthermore, certain elementary columns are inserted into $\widetilde{\mathbf{H}}$ resulting in a square nonsingular matrix $\mathcal{L}_{\vec{\delta}}(\mathbf{H})$ which preserves fundamental properties of $\mathbf{H}$ (for example, its Smith factors and its determinant). The

matrix $\mathcal{L}_{\vec{\delta}}(\mathbf{H})$ has dimension $\widetilde{n} \times \widetilde{n}$ and degree at most δ, which in this case is the average row degree of $\mathbf{H}$.

Consider for example a 4×4 matrix $\mathbf{H}$ in Hermite form with diagonal entries having degrees $(2, 37, 7, 18)$. Such a matrix has degree profile

$$\mathbf{H} = \begin{bmatrix} (2) & & & \\ [36] & (37) & & \\ [6] & [6] & (7) & \\ [17] & [17] & [17] & (18) \end{bmatrix},$$

where $[d]$ stands for an entry of degree at most d and (d) stands for a monic entry of degree exactly d. Here $\mathbf{H}$ has row degree $\vec{\delta} = (2, 37, 7, 18)$.

Let us now construct the row partial linearization $\mathcal{L}_{\vec{\delta}}(\mathbf{H})$. Considering the upper bound $\delta = 1 + \lfloor (2+37+7+18)/4 \rfloor = 17$ on the average row degree of $\mathbf{H}$, we will split the high-degree rows of $\mathbf{H}$ in several rows having degree less than δ. The first row is unchanged; the second row is expanded into two rows of degree 16 and one row of degree 3; the third row is unchanged; and finally the last row is expanded into one row of degree 16 and one row of degree 1. The matrix with expanded rows is then

$$\widetilde{\mathbf{H}} = \begin{bmatrix} (2) & & & \\ [16] & [16] & & \\ [16] & [16] & & \\ [2] & (3) & & \\ [6] & [6] & (7) & \\ [16] & [16] & [16] & [16] \\ [0] & [0] & [0] & (1) \end{bmatrix}.$$

Note that $\mathbf{H}$ and $\widetilde{\mathbf{H}}$ are related by $\boldsymbol{\mathcal{E}}_{\vec{\delta}} \cdot \widetilde{\mathbf{H}} = \mathbf{H}$, where $\boldsymbol{\mathcal{E}}_{\vec{\delta}}$ is the so-called *expansion-compression* matrix

$$\boldsymbol{\mathcal{E}}_{\vec{\delta}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & x^{17} & x^{34} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & x^{17} \end{bmatrix}.$$

We can insert elementary columns in $\widetilde{\mathbf{H}}$ by

$$\mathcal{L}_{\vec{\delta}}(\mathbf{H}) = \begin{bmatrix} (2) & & & & & & \\ [16] & x^{17} & & [16] & & & \\ [16] & -1 & x^{17} & [16] & & & \\ [2] & & -1 & (3) & & & \\ [6] & & & [6] & (7) & & \\ [16] & & & [16] & [16] & x^{17} & [16] \\ [0] & & & [0] & [0] & -1 & (1) \end{bmatrix}$$

which indicate the row operations needed to keep track of the structure of the original rows of $\mathbf{H}$. Now the reduced tuple of row degrees $\vec{d} = (2, 17, 17, 3, 7, 17, 1)$ has as its largest entry the *average* row degree $\delta = 17$ of $\mathbf{H}$. Furthermore, $\mathbf{H}$ can be reconstructed from $\mathcal{L}_{\vec{\delta}}(\mathbf{H})$, without field operations, as a submatrix of $\boldsymbol{\mathcal{E}}_{\vec{\delta}} \cdot \mathcal{L}_{\vec{\delta}}(\mathbf{H})$.

Remark 5.4. This partial linearization differs from that of [15, Theorem 10] in that

- it operates on the rows rather than the columns,
- it scales the inserted elementary columns by -1 compared to the elementary rows in [15], and
- it keeps the linearized rows together. This reflects the fact that the expansion-compression matrix $\boldsymbol{\mathcal{E}}_{\vec{\delta}}$ is a column permutation of the one in the construction of [15], which would be

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & x^{17} & x^{34} & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & x^{17} \end{bmatrix}$$

  in the example above.

Our motivation for these changes is that the partial row linearization we use here preserves shifted reduced and shifted Popov forms. This will be detailed below. ◇

Formally we define the partial linearization for a matrix $\mathbf{A}$ and a tuple $\vec{\delta}$, with the latter not necessarily related to $\operatorname{rdeg}(\mathbf{A})$. Indeed, we will apply this in a situation where the tuple $\vec{\delta}$ is formed by the diagonal degrees of the Hermite form of $\mathbf{A}$.

Definition 5.5. *Let* $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$, $\vec{\delta} = (\delta_1, \ldots, \delta_n) \in \mathbb{N}^n$ *and set*

$$\delta = 1 + \left\lfloor \frac{(\delta_1 + \cdots + \delta_n)}{n} \right\rfloor.$$

For any $i \in \{1, \ldots, n\}$ *write* $\delta_i = (\alpha_i - 1)\delta + \beta_i$ *with* $\alpha_i = \lceil \delta_i/\delta \rceil$ *and* $1 \le \beta_i \le \delta$ *if* $\delta_i > 0$, *while* $\alpha_i = 1$ *and* $\beta_i = 0$ *if* $\delta_i = 0$. *Set* $\widetilde{n} = \alpha_1 + \cdots + \alpha_n$ *and define* $\vec{d} \in \mathbb{N}^{\widetilde{n}}$ *as*

$$\vec{d} = (\underbrace{\delta, \ \ldots\ , \ \delta, \ \beta_1}_{\alpha_1}, \ \ldots\ , \ \underbrace{\delta, \ \ldots, \ \delta, \ \beta_n}_{\alpha_n}) \tag{6}$$

as well as the row expansion-compression matrix $\boldsymbol{\mathcal{E}}_{\vec{\delta}} \in \mathbb{K}[x]^{n\times\widetilde{n}}$ *as*

$$\boldsymbol{\mathcal{E}}_{\vec{\delta}} = \begin{bmatrix} 1 & x^{\delta} & \cdots & x^{(\alpha_1-1)\delta} & & & & \\ & & & & \ddots & & & \\ & & & & & 1 & x^{\delta} & \cdots & x^{(\alpha_n-1)\delta} \end{bmatrix}. \tag{7}$$

Let $\widetilde{\mathbf{A}} \in \mathbb{K}[x]^{\widetilde{n}\times n}$ *be such that* $\mathbf{A} = \boldsymbol{\mathcal{E}}_{\vec{\delta}} \cdot \widetilde{\mathbf{A}}$ *with all the rows of* $\widetilde{\mathbf{A}}$ *having degree at most* δ *except possibly at indices* $\{\alpha_1 + \cdots + \alpha_i, 1 \le i \le n\}$. *Define* $\mathcal{L}_{\vec{\delta}}(\mathbf{A}) \in \mathbb{K}[x]^{\widetilde{n}\times\widetilde{n}}$ *as:*

(i) for $1 \le i \le n$, *the column* $\alpha_1 + \cdots + \alpha_i$ *of* $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ *is the column* i *of* $\widetilde{\mathbf{A}}$;

(ii) for $0 \le i \le n-1$ *and* $1 \le j \le \alpha_{i+1} - 1$, *the column* $\alpha_1 + \cdots + \alpha_i + j$ *of* $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ *is the column*

$$[0, \cdots, 0, x^{\delta}, -1, 0, \cdots, 0]^{\mathsf{T}} \in \mathbb{K}[x]^{\widetilde{n}\times 1}$$

with the entry x^{δ} *at row index* $\alpha_1 + \cdots + \alpha_i + j$.

It follows from this construction that any matrix $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$ is the submatrix of $\boldsymbol{\mathcal{E}}_{\vec{\delta}} \cdot \mathcal{L}_{\vec{\delta}}(\mathbf{A})$ formed by its columns at indices $\{\alpha_1 + \cdots + \alpha_i, 1 \le i \le n\}$.

It is important to note that this transformation has good properties regarding the computation of $-\vec{\delta}$-shifted reduced forms of $\mathbf{A}$, where $\vec{\delta}$ is the tuple of diagonal degrees of the Hermite form of $\mathbf{A}$. Indeed, it transforms any $-\vec{\delta}$-reduced form $\mathbf{R}$ of $\mathbf{A}$ into a $-\vec{d}$-reduced form $\mathcal{L}_{\vec{\delta}}(\mathbf{R})$ of the transformed $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$. In other words, we have the following diagram:

$$\begin{array}{ccc} \mathbf{x}^{\vec{\mu}-\vec{\delta}}\,\mathbf{A} & \xrightarrow{\text{reduction}} & -\vec{\delta}\text{-reduced form of } \mathbf{A} \\ \Big| & & \Big| \\ \text{partial linearization} & & \text{partial linearization} \\ \downarrow & & \downarrow \\ \mathbf{x}^{\vec{m}-\vec{d}}\,\mathcal{L}_{\vec{\delta}}(\mathbf{A}) & \xrightarrow{\text{reduction}} & -\vec{d}\text{-reduced form of } \mathcal{L}_{\vec{\delta}}(\mathbf{A}) \end{array},$$

where $\vec{m}$ is the uniform tuple $(\max(\vec{d}), \ldots, \max(\vec{d}))$ of length $\widetilde{n}$. In terms of efficiency, it is more interesting to perform the reduction step on $\mathbf{x}^{\vec{m}-\vec{d}}\,\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ with the shift $-\vec{d}$, rather than on $\mathbf{A}$ with the shift $-\vec{\delta}$. Indeed, using the fastest known deterministic reduction algorithm [15], the latter computation uses $\widetilde{\mathcal{O}}(n^\omega(\deg(\mathbf{A}) + \max(\vec{\delta})))$ field operations. On the other hand, the former is in $\widetilde{\mathcal{O}}(n^\omega(\deg(\mathbf{A}) + \delta))$, since $\max(\vec{d}) \le \delta$ and $\deg(\mathcal{L}_{\vec{\delta}}(\mathbf{A})) \le \deg(\mathbf{A})$. We recall that δ is close to the average of $\vec{\delta}$.

We state this formally in the following lemma. For the sake of presentation we postpone the proof until later in Section 5.4.

Lemma 5.6. *Let $\vec{\delta} = (\delta_1, \ldots, \delta_n) \in \mathbb{N}^n$, and define $\vec{d}$ as in equation (6).*

(i) If a matrix $\mathbf{R} \in \mathbb{K}[x]^{n\times n}$ is $-\vec{\delta}$-reduced with $-\vec{\delta}$-column degree $\vec{0}$, then $\mathcal{L}_{\vec{\delta}}(\mathbf{R})$ is $-\vec{d}$-reduced with $-\vec{d}$-column degree $\vec{0}$.

(ii) If two matrices $\mathbf{A}$ and $\mathbf{B}$ in $\mathbb{K}[x]^{n\times n}$ are right unimodularly equivalent, then $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ and $\mathcal{L}_{\vec{\delta}}(\mathbf{B})$ are also right unimodularly equivalent.

(iii) If $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$ is nonsingular, $\mathbf{R}$ is a $-\vec{\delta}$-reduced form of $\mathbf{A}$, and $\mathbf{R}$ has $-\vec{\delta}$-column degree $\vec{0}$, then $\mathcal{L}_{\vec{\delta}}(\mathbf{R})$ is a $-\vec{d}$-reduced form of $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ with $-\vec{d}$-column degree $\vec{0}$.

Our algorithm will first build $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ and then find a $-\vec{d}$-reduced form $\hat{\mathbf{R}}$ for this new matrix. We note that, for any $-\vec{\delta}$-reduced form $\mathbf{R}$ of $\mathbf{A}$, the matrix $\hat{\mathbf{R}} = \mathcal{L}_{\vec{\delta}}(\mathbf{R})$ is a suitable reduced form and, as remarked earlier, has the property that it is easy to recover $\mathbf{R}$. However, it is not the case that any $\hat{\mathbf{R}}$ computed by shifted reduction from $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ will have the form $\hat{\mathbf{R}} = \mathcal{L}_{\vec{\delta}}(\mathbf{R})$. In order to solve this issue, we will rely on normalization as in Lemma 5.2. This allows us to deduce $\mathcal{L}_{\vec{\delta}}(\mathbf{H})$ from $\hat{\mathbf{R}}$, and then the entries of $\mathbf{H}$ can be read off from those of $\mathcal{L}_{\vec{\delta}}(\mathbf{H})$. Diagrammatically we have

$$\begin{array}{ccccc} \mathbf{x}^{\vec{\mu}-\vec{\delta}}\,\mathbf{A} & \xrightarrow{\text{reduction}} & \mathbf{x}^{\vec{\mu}-\vec{\delta}}\,\mathbf{R} & \xrightarrow{\text{normalization}} & \mathbf{H} = \mathbf{R} \cdot \mathrm{lm}_{-\vec{\delta}}(\mathbf{R})^{-1} \\ \Big| & & & & \Big| \\ \text{partial linearization} & & & & \text{partial linearization} \\ \downarrow & & & & \downarrow \\ \mathbf{x}^{\vec{m}-\vec{d}}\,\mathcal{L}_{\vec{\delta}}(\mathbf{A}) & \xrightarrow{\text{reduction}} & \mathbf{x}^{\vec{m}-\vec{d}}\,\hat{\mathbf{R}} & \xrightarrow{\text{normalization}} & \mathcal{L}_{\vec{\delta}}(\mathbf{H}) = \hat{\mathbf{R}} \cdot \mathrm{lm}_{-\vec{d}}(\hat{\mathbf{R}})^{-1} \end{array}.$$

Corollary 5.7. *Let $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$ be nonsingular and let $\vec{\delta} = (\delta_1, \ldots, \delta_n) \in \mathbb{N}^n$ denote the degrees of the diagonal entries of the Hermite form $\mathbf{H}$ of $\mathbf{A}$. Using the notation from Definition 5.5, we have that*

(i) $\mathrm{lm}_{-\vec{d}}(\mathcal{L}_{\vec{\delta}}(\mathbf{H}))$ *is the identity matrix,*

(ii) if $\hat{\mathbf{R}} \in \mathbb{K}[x]^{\widetilde{n}\times\widetilde{n}}$ *is a* $-\vec{d}$*-reduced form of* $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$*, then* $\mathcal{L}_{\vec{\delta}}(\mathbf{H}) = \hat{\mathbf{R}} \cdot \mathrm{lm}_{-\vec{d}}(\hat{\mathbf{R}})^{-1}$.

Proof. (i) follows from the construction of $\mathcal{L}_{\vec{\delta}}(\mathbf{H})$. From Lemma 5.6 we have that $\mathcal{L}_{\vec{\delta}}(\mathbf{H})$ is a $-\vec{d}$-reduced form of $\mathbf{A}$, so that (ii) follows from (i) and Lemma 5.2. □

In particular, $\mathbf{H}$ can be recovered as being the submatrix of $\boldsymbol{\mathcal{E}}_{\vec{\delta}} \cdot \hat{\mathbf{R}}\,\mathrm{lm}_{-\vec{d}}(\hat{\mathbf{R}})^{-1}$ formed by its columns $\{\alpha_1 + \cdots + \alpha_i, 1 \le i \le n\}$.

Example 5.8 (Reducing the diagonal degrees). Consider a matrix $\mathbf{A} \in \mathbb{K}[x]^{4\times4}$ such that its Hermite form $\mathbf{H}$ has diagonal degrees $\vec{\delta} = (2, 37, 7, 18)$. As shown earlier,

$$\mathcal{L}_{\vec{\delta}}(\mathbf{H}) = \begin{bmatrix} (2) & & & & & & \\ [16] & x^{17} & & [16] & & & \\ [16] & -1 & x^{17} & [16] & & & \\ [2] & & -1 & (3) & & & \\ [6] & & & [6] & (7) & & \\ [16] & & & [16] & [16] & x^{17} & [16] \\ [0] & & & [0] & [0] & -1 & (1) \end{bmatrix}.$$

We see that $\vec{d} = (2, 17, 17, 3, 7, 17, 1)$ corresponds to the row degree of $\mathcal{L}_{\vec{\delta}}(\mathbf{H})$, that this matrix has $-\vec{d}$-column degree $\vec{0}$ and that its $-\vec{d}$-leading matrix is the identity. In particular, it is $-\vec{d}$-reduced. In addition, from (ii) of Lemma 5.6, $\mathcal{L}_{\vec{\delta}}(\mathbf{H})$ and $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ are right-unimodularly equivalent. As a result, $\mathcal{L}_{\vec{\delta}}(\mathbf{H})$ is a $-\vec{d}$-reduced form of $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$.

Let $\hat{\mathbf{R}}$ be any $-\vec{d}$-reduced form of $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$. Then $\hat{\mathbf{R}}$ also has $-\vec{d}$-column degree $\vec{0}$, its $-\vec{d}$-leading matrix is invertible, and its degree profile is

$$\hat{\mathbf{R}} = \begin{bmatrix} [2] & [2] & [2] & [2] & [2] & [2] & [2] \\ [17] & [17] & [17] & [17] & [17] & [17] & [17] \\ [17] & [17] & [17] & [17] & [17] & [17] & [17] \\ [3] & [3] & [3] & [3] & [3] & [3] & [3] \\ [7] & [7] & [7] & [7] & [7] & [7] & [7] \\ [17] & [17] & [17] & [17] & [17] & [17] & [17] \\ [1] & [1] & [1] & [1] & [1] & [1] & [1] \end{bmatrix}.$$

While $\hat{\mathbf{R}}$ is generally not of the form $\mathcal{L}_{\vec{\delta}}(\mathbf{R})$ for $\mathbf{R}$ some $-\vec{\delta}$-reduced form of $\mathbf{A}$, it still follows from Lemma 5.2 that $\mathcal{L}_{\vec{\delta}}(\mathbf{H}) = \hat{\mathbf{R}} \cdot \mathrm{lm}_{-\vec{d}}(\hat{\mathbf{R}})^{-1}$. ◇

5.3. Algorithm and computational cost

The results in the previous subsection lead to Algorithm 3 for the computation of the Hermite form $\mathbf{H}$ from $\mathbf{A}$ and $\vec{\delta}$. Its main computational task is to compute a column reduced form of a matrix of dimension $\mathcal{O}(n)$ and degree $\mathcal{O}(\deg(\mathbf{A}))$ (Step 12). This can be done efficiently and deterministically with the algorithm in [15, Section 8].

Algorithm 3 HermiteKnownDegree($\mathbf{A}, \vec{s}, \vec{\delta}$)

Input: $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$ a nonsingular matrix, $\vec{\delta} = (\delta_1, \ldots, \delta_n) \in \mathbb{N}^n$ the degrees of the diagonal entries of the Hermite form of $\mathbf{A}$.

Output: the Hermite form of $\mathbf{A}$.

1: $\delta := 1 + \lfloor(\delta_1 + \cdots + \delta_n)/n\rfloor$;
2: **for** $i \in \{1, \ldots, n\}$ **do**
3: **if** $\delta_i > 0$ **then**
4: $\alpha_i := \lceil \delta/\delta_i \rceil$; $\beta_i := \delta_i - (\alpha_i - 1)\delta$;
5: **else**
6: $\alpha_i := 1$; $\beta_i = 0$;
7: **end if**
8: **end for**
9: $\widetilde{n} := \alpha_1 + \cdots + \alpha_n$ and $\boldsymbol{\mathcal{E}}_{\vec{\delta}} \in \mathbb{K}^{\widetilde{n}\times n}$ as in equation (7);
10: $\vec{d} = (d_1, \ldots, d_{\widetilde{n}})$ as in equation (6);
11: $\mathbf{D} := \mathrm{Diag}(x^{\delta-d_1}, \ldots, x^{\delta-d_{\widetilde{n}}})$;
12: $\mathbf{D}\hat{\mathbf{R}}$:= column reduced form of $\mathbf{D} \cdot \mathcal{L}_{\vec{\delta}}(\mathbf{A})$; {*using the algorithm in [15]*}
13: $\hat{\mathbf{H}} := \boldsymbol{\mathcal{E}}_{\vec{\delta}} \cdot \hat{\mathbf{R}} \cdot \mathrm{lm}_{-\vec{d}}(\hat{\mathbf{R}})^{-1}$;
14: $\mathbf{H}$:= the submatrix of $\hat{\mathbf{H}}$ formed by its columns $\{\alpha_1 + \cdots + \alpha_i, 1 \le i \le n\}$
15: **return** $\mathbf{H}$;

Proposition 5.9. *Let $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$ be nonsingular, and let $\vec{\delta} \in \mathbb{N}^n$ be the degrees of the diagonal entries of the Hermite form of $\mathbf{A}$. On input $\mathbf{A}$ and $\vec{\delta}$, Algorithm 3 computes the Hermite form of $\mathbf{A}$ using $\widetilde{\mathcal{O}}(n^\omega \deg(\mathbf{A}))$ field operations.*

Proof. The correctness of the algorithm follows directly from Corollary 5.7 and from the remark that a matrix $\mathbf{R} \in \mathbb{K}[x]^{\widetilde{n}\times\widetilde{n}}$ is $-\vec{d}$-column reduced if and only if $\mathbf{D} \cdot \mathbf{R}$ is column reduced (for the uniform shift), where $\mathbf{D}$ is the diagonal matrix at Step 11.

Furthermore, we have $\deg(\mathbf{D}) \le \delta$ and $\deg(\mathcal{L}_{\vec{\delta}}(\mathbf{A})) \le \max(\deg(\mathbf{A}), \delta)$. Since $\delta = 1+\lfloor|\vec{\delta}|/n\rfloor$, and as $\mathbf{H}$ is in Hermite form and $\vec{\delta}$ are the degrees of its diagonal entries, we have $|\vec{\delta}| = \deg(\det(\mathbf{H})) = \deg(\det(\mathbf{A})) \le n\deg(\mathbf{A})$. Thus, $\delta \le 1 + \deg(\mathbf{A})$ and the degrees of $\mathbf{D}$ and $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ are both at most $1 + \deg(\mathbf{A})$. Their product $\mathbf{D} \cdot \mathcal{L}_{\vec{\delta}}(\mathbf{A})$ therefore has degree at most $2 + 2\deg(\mathbf{A})$. On the other hand, these matrices have dimension

$$\widetilde{n} = \sum_{i=1}^{n} \alpha_i \le \sum_{i=1}^{n} (1 + \delta_i/\delta) = n + \frac{|\vec{\delta}|}{1 + \lfloor|\vec{\delta}|/n\rfloor} < 2n.$$

As a result, Step 12 uses $\widetilde{\mathcal{O}}(n^\omega \deg(\mathbf{A}))$ field operations [15, Theorem 18].

Concerning Step 13, from Corollary 5.7 the matrix $\hat{\mathbf{R}}$ has row degree $\vec{d}$, and $\mathrm{lm}_{-\vec{d}}(\hat{\mathbf{R}})^{-1}$ is a constant matrix. Thus the computation of $\hat{\mathbf{R}} \cdot \mathrm{lm}_{-\vec{d}}(\hat{\mathbf{R}})^{-1}$ can be performed via complete linearization of the rows of $\hat{\mathbf{R}}$, using $\mathcal{O}(n^\omega \lceil|\vec{d}|/n\rceil)$ operations. This concludes the proof since $|\vec{d}| = |\vec{\delta}| = \deg(\det(\mathbf{H})) = \deg(\det(\mathbf{A})) \le n\deg(\mathbf{A})$. □

Combining Algorithms 1 and 3 results in a deterministic algorithm for computing the Hermite form of $\mathbf{A}$ in $\widetilde{\mathcal{O}}(n^\omega \deg(\mathbf{A}))$ field operations.

Example 5.10. Let $\mathbb{K} = \mathbb{Z}_7$ be the field with 7 elements, and consider the matrix $\mathbf{A} \in \mathbb{K}[x]^{3\times 3}$ from Example 3.2:

$$\mathbf{A} = \begin{bmatrix} 6x+1 & 2x^3+x^2+6x+1 & 3 \\ 4x^5+5x^4+4x^2+x & 6x^5+5x^4+2x^3+4 & x^4+5x^3+6x^2+5x \\ 2 & 2x^5+5x^4+5x^3+6x^2 & 6 \end{bmatrix}.$$

According to Example 3.2 the diagonal entries of the Hermite form of $\mathbf{A}$ have degrees $\vec{\delta} = (0, 1, 9)$. Note that $\vec{\delta}$ is non-uniform, and $\max(\vec{\delta}) - \min(\vec{\delta}) = \deg(\det(\mathbf{A})) - 1$.

Using the column reduction algorithm in [15] to compute a $-\vec{\delta}$-reduced form of $\mathbf{A}$ would imply working on the matrix $\mathbf{x}^{\vec{\mu}-\vec{\delta}}\,\mathbf{A} = \mathbf{x}^{(9,8,0)}\,\mathbf{A}$, which has degree $13 = \deg(\det(\mathbf{A})) + \deg(\mathbf{A}) - 2$. In this case partial linearization gives us a 5×5 matrix $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ and a shift $\vec{d}$ such that $\deg(\mathcal{L}_{\vec{\delta}}(\mathbf{A})) \leq \deg(\mathbf{A})$ and $\max(\vec{d}) - \min(\vec{d}) \leq \deg(\mathbf{A})$. In particular, the matrix $\mathbf{x}^{\vec{m}-\vec{d}}\,\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ to be reduced has degree $8 \leq 2\deg(\mathbf{A})$.

To see this, Definition 5.5 gives the parameters $\delta = 4$, $\vec{\alpha} = (1, 1, 3)$, $\vec{\beta} = (0, 1, 1)$, $\vec{d} = (0, 1, 4, 4, 1)$, the expansion-compression matrix

$$\boldsymbol{\mathcal{E}}_{\vec{\delta}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & x^4 & x^8 \end{bmatrix},$$

and finally

$$\mathcal{L}_{\vec{\delta}}(\mathbf{A}) = \begin{bmatrix} 6x+1 & 2x^3+x^2+6x+1 & 0 & 0 & 3 \\ 4x^5+5x^4+4x^2+x & 6x^5+5x^4+2x^3+4 & 0 & 0 & x^4+5x^3+6x^2+5x \\ 2 & 5x^3+6x^2 & x^4 & 0 & 6 \\ 0 & 2x+5 & 6 & x^4 & 0 \\ 0 & 0 & 0 & 6 & 0 \end{bmatrix}.$$

Computing a $-\vec{d}$-reduced form for $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ gives

$$\hat{\mathbf{R}} = \begin{bmatrix} 5 & 1 & 0 & 1 & 2 \\ 5 & 4x+4 & 0 & 3x+5 & 6x+3 \\ x^3+6x^2+4 & 3x^4+x^3+6x^2 & x^4 & x^3+5x^2+4x+3 & 6x^4+2x^3+3x^2+x+6 \\ 3x^3+4x^2+6 & 4x^4+4x^3+4x+5 & 6 & x^3+2x+4 & 5x^4+2x^3+4x+2 \\ 6 & x & 0 & 6 & 0 \end{bmatrix}.$$

Note that $\operatorname{rdeg}(\mathbf{R}) = \vec{d}$, and more precisely,

$$\operatorname{lm}_{-\vec{d}}(\mathbf{R}) = \begin{bmatrix} 5 & 1 & 0 & 1 & 2 \\ 0 & 4 & 0 & 3 & 6 \\ 0 & 3 & 1 & 0 & 6 \\ 0 & 4 & 0 & 0 & 5 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

Normalizing $\hat{\mathbf{R}}$ via $\hat{\mathbf{R}} \cdot \mathrm{lm}_{-\vec{d}}(\mathbf{R})^{-1}$ gives

$$\mathcal{L}_{\vec{\delta}}(\mathbf{H}) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & x+6 & 0 & 0 & 0 \\ 3x^3+4x^2+5 & 4x^3+5x^2+6x+4 & x^4 & 0 & 3x^3+3x^2+4x \\ 2x^3+5x^2+4 & 2x^3+3x^2+3x & 6 & x^4 & x^3+4x^2+6x+4 \\ 4 & 3 & 0 & 6 & x+2 \end{bmatrix}.$$

Performing the inverse linearization, by taking columns $(1,2,5)$ of $\boldsymbol{\mathcal{E}}_{\vec{\delta}} \cdot \mathcal{L}_{\vec{\delta}}(\mathbf{H})$, directly gives the entries in the Hermite form of $\mathbf{A}$:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & x+6 & 0 \\ h_{31} & h_{32} & x^9+2x^8+x^7+4x^6+6x^5+4x^4+3x^3+3x^2+4x \end{bmatrix}$$

with

$$\begin{aligned} h_{31} &= 4x^8+2x^7+5x^6+4x^4+3x^3+4x^2+5, \\ h_{32} &= 3x^8+2x^7+3x^6+3x^5+4x^3+5x^2+6x+4. \quad \diamond \end{aligned}$$

5.4. Proof of Lemma 5.6

Let us now give the detailed proof of Lemma 5.6.

(i) Since $\mathbf{R} \in \mathbb{K}[x]^{n\times n}$ is $-\vec{\delta}$-reduced with $-\vec{\delta}$-column degree $\vec{0}$, it has row degree $\vec{\delta}$ since otherwise the invertible matrix $\mathrm{lm}_{-\vec{\delta}}(\mathbf{R})$ would have a zero row. We show that $\mathrm{lm}_{-\vec{d}}(\mathcal{L}_{\vec{\delta}}(\mathbf{R}))$ is a permutation of the rows and columns of $\begin{bmatrix} \mathrm{lm}_{-\vec{\delta}}(\mathbf{R}) & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \in \mathbb{K}^{\widetilde{n}\times\widetilde{n}}$. In particular, $\mathrm{lm}_{-\vec{d}}(\mathcal{L}_{\vec{\delta}}(\mathbf{R}))$ is invertible and thus $\mathcal{L}_{\vec{\delta}}(\mathbf{R})$ is $-\vec{d}$-reduced.

Let us first observe it on an example. We consider the case $\vec{\delta} = (2, 37, 7, 18)$. Then $\mathbf{R}$ has the following degree profile,

$$\mathbf{R} = \begin{bmatrix} [2] & [2] & [2] & [2] \\ [37] & [37] & [37] & [37] \\ [7] & [7] & [7] & [7] \\ [18] & [18] & [18] & [18] \end{bmatrix}$$

with invertible $-\vec{\delta}$-leading matrix. Following the construction in Definition 5.5, we have $\vec{d} = (2, 17, 17, 3, 7, 17, 1)$ and

$$\mathcal{L}_{\vec{\delta}}(\mathbf{R}) = \begin{bmatrix} [2] & & & [2] & [2] & & [2] \\ [16] & x^{17} & & [16] & [16] & & [16] \\ [16] & -1 & x^{17} & [16] & [16] & & [16] \\ [3] & & -1 & [3] & [3] & & [3] \\ [7] & & & [7] & [7] & & [7] \\ [16] & & & [16] & [16] & x^{17} & [16] \\ [1] & & & [1] & [1] & -1 & [1] \end{bmatrix}.$$

Observe that $\mathbf{R}$ has $-\vec{d}$-column degree at most $\vec{0}$ componentwise, and that its $-\vec{d}$-leading matrix is

$$\mathrm{lm}_{-\vec{d}}(\mathcal{L}_{\vec{\delta}}(\mathbf{R})) = \begin{bmatrix} \ell_{11} & & & \ell_{12} & \ell_{13} & & \ell_{14} \\ & 1 & & & & & \\ & & 1 & & & & \\ \ell_{21} & & & \ell_{22} & \ell_{23} & & \ell_{24} \\ \ell_{31} & & & \ell_{32} & \ell_{33} & & \ell_{34} \\ & & & & & 1 & \\ \ell_{41} & & & \ell_{42} & \ell_{42} & & \ell_{42} \end{bmatrix},$$

where $(\ell_{ij})_{1\leq i,j\leq 4} = \mathrm{lm}_{-\vec{\delta}}(\mathbf{R})$. Since $\mathrm{lm}_{-\vec{\delta}}(\mathbf{R})$ is invertible, $\mathrm{lm}_{-\vec{d}}(\mathcal{L}_{\vec{\delta}}(\mathbf{R}))$ is invertible as well. Furthermore $\mathcal{L}_{\vec{\delta}}(\mathbf{R})$ is $-\vec{d}$-reduced and that it has $-\vec{d}$-column degree $\vec{0}$.

In the general case, by construction of $\mathcal{L}_{\vec{\delta}}(\mathbf{R})$ one can check that $\mathrm{lm}_{-\vec{d}}(\mathcal{L}_{\vec{\delta}}(\mathbf{R}))$ is a matrix in $\mathbb{K}^{\widetilde{n}\times\widetilde{n}}$ such that

(a) its $n \times n$ submatrix with row and column indices in $\{\alpha_1 + \cdots + \alpha_i, 1 \leq i \leq n\}$ is equal to $\mathrm{lm}_{-\vec{\delta}}(\mathbf{R})$,

(b) its $(\widetilde{n}-n)\times(\widetilde{n}-n)$ submatrix with row and column indices in $\{1,\ldots,\widetilde{n}\} - \{\alpha_1+\cdots+\alpha_i, 1 \leq i \leq n\}$ is equal to the identity matrix,

(c) its other entries are all zero.

This directly implies that $\mathrm{lm}_{-\vec{d}}(\mathcal{L}_{\vec{\delta}}(\mathbf{R}))$ is invertible. In addition by construction $\mathcal{L}_{\vec{\delta}}(\mathbf{R})$ has $-\vec{d}$-column degree at most $\vec{0}$ componentwise. The fact that $\mathrm{lm}_{-\vec{d}}(\mathcal{L}_{\vec{\delta}}(\mathbf{R}))$ is invertible also implies that $\mathcal{L}_{\vec{\delta}}(\mathbf{R})$ has $-\vec{d}$-column degree exactly $\vec{0}$.

(*ii*) Denote by $\boldsymbol{\mathcal{T}}_{\vec{\delta}} \in \mathbb{K}[x]^{\widetilde{n}\times\widetilde{n}-n}$ the submatrix of $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ formed by its columns at indices $\{\alpha_1 + \cdots + \alpha_i + j, 1 \leq j \leq \alpha_{i+1} - 1, 0 \leq i \leq n-1\}$. Up to a permutation of its columns, $\mathcal{L}_{\vec{\delta}}(\mathbf{A})$ is then $[\boldsymbol{\mathcal{T}}_{\vec{\delta}} \;\; \widetilde{\mathbf{A}}]$. In particular, $\boldsymbol{\mathcal{E}}_{\vec{\delta}} \cdot \mathcal{L}_{\vec{\delta}}(\mathbf{A})$ is right-unimodularly equivalent to $\boldsymbol{\mathcal{E}}_{\vec{\delta}}[\boldsymbol{\mathcal{T}}_{\vec{\delta}} \;\; \widetilde{\mathbf{A}}] = [\mathbf{0} \;\; \mathbf{A}]$. For the remainder of this proof we will use the shorthand notation $\boldsymbol{\mathcal{E}}_{\vec{\delta}} \cdot \mathcal{L}_{\vec{\delta}}(\mathbf{A}) \equiv [\mathbf{0} \;\; \mathbf{A}]$.

Define the matrix $\mathbf{E} \in \mathbb{K}^{(\widetilde{n}-n)\times\widetilde{n}}$ whose row $\alpha_1 + \cdots + \alpha_i + j - i$ is the coordinate vector with 1 at index $\alpha_1 + \cdots + \alpha_i + j + 1$, for all $1 \leq j \leq \alpha_{i+1} - 1$ and $0 \leq i \leq n-1$. That is, we have

$$\begin{bmatrix} \mathbf{E} \\ \boldsymbol{\mathcal{E}}_{\vec{\delta}} \end{bmatrix} = \begin{bmatrix} 0 & 1 & & & & & & & & & & & \\ & & \ddots & & & & & & & & & & \\ & & & & 1 & & & & & & & & \\ & & & & & & \ddots & & & & & & \\ & & & & & & & 0 & 1 & & & & \\ & & & & & & & & & \ddots & & & \\ & & & & & & & & & & & 1 & \\ 1 & x^{\delta} & \cdots & x^{(\alpha_1-1)\delta} & & & & & & & & & \\ & & & & & & \ddots & & & & & & \\ & & & & & & & 1 & x^{\delta} & \cdots & x^{(\alpha_n-1)\delta} \end{bmatrix}.$$

By construction, the matrix $\mathbf{U} := \mathbf{E} \cdot \boldsymbol{\mathcal{T}}_{\vec{\delta}}$ is upper triangular with diagonal entries -1, and thus

unimodular. As a result,

$$\begin{bmatrix} \mathbf{E} \\ \boldsymbol{\mathcal{E}}_{\vec{\delta}} \end{bmatrix} \mathcal{L}_{\vec{\delta}}(\mathbf{A}) \equiv \begin{bmatrix} \mathbf{E} \\ \boldsymbol{\mathcal{E}}_{\vec{\delta}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\mathcal{T}}_{\vec{\delta}} & \widetilde{\mathbf{A}} \end{bmatrix} = \begin{bmatrix} \mathbf{U} & * \\ \mathbf{0} & \mathbf{A} \end{bmatrix} \equiv \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{A} \end{bmatrix}.$$

Similarly, we have that $\begin{bmatrix} \mathbf{E} \\ \boldsymbol{\mathcal{E}}_{\vec{\delta}} \end{bmatrix} \mathcal{L}_{\vec{\delta}}(\mathbf{B}) \equiv \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{bmatrix}$.

Since $\mathbf{A} \equiv \mathbf{B}$ by assumption, we obtain $\begin{bmatrix} \mathbf{E} \\ \boldsymbol{\mathcal{E}}_{\vec{\delta}} \end{bmatrix} \mathcal{L}_{\vec{\delta}}(\mathbf{A}) \equiv \begin{bmatrix} \mathbf{E} \\ \boldsymbol{\mathcal{E}}_{\vec{\delta}} \end{bmatrix} \mathcal{L}_{\vec{\delta}}(\mathbf{B})$. This implies that $\mathcal{L}_{\vec{\delta}}(\mathbf{A}) \equiv \mathcal{L}_{\vec{\delta}}(\mathbf{B})$ since the matrix $\begin{bmatrix} \mathbf{E} \\ \boldsymbol{\mathcal{E}}_{\vec{\delta}} \end{bmatrix}$ is invertible (more precisely, its determinant is 1).

(iii) is a direct consequence of (i) and (ii).

6. Reduction to almost uniform degrees in Hermite form computation

As mentioned in Section 2.3, we aim at a cost bound which involves the generic determinant bound. In Section 3 we showed how to compute the diagonal entries of $\mathbf{H}$ in $\widetilde{\mathcal{O}}(n^\omega \lceil s \rceil)$ operations, with s the average column degree of the input matrix. However, this does not take into account the fact that the degrees of its rows are possibly unbalanced. In Section 5, we were only able to obtain the cost bound $\widetilde{\mathcal{O}}(n^\omega \deg(\mathbf{A}))$ for computing the remaining entries of $\mathbf{H}$.

The goal of this section is to show that, applying results from [15, Section 6], one can give a reduction from the general case of Hermite form computation to the case where the degree of the input matrix $\mathbf{A}$ is in $\mathcal{O}(\lceil D(\mathbf{A})/n \rceil)$. This is stated formally in Proposition 6.1, after what we give two complete examples to illustrate this reduction (Examples 6.2 and 6.3).

To get a rough idea of how the partial linearization in [15, Section 6] works and how it benefits Hermite form computation, consider the matrix

$$\mathbf{A} = \begin{bmatrix} 1 & x^{39} + x \\ x & x^{41} + 1 \end{bmatrix}.$$

In this case the column degrees of the matrix are quite unbalanced as 1 and 41 have an average column degree of 21. However we can create a second matrix, of slightly larger dimension, as

$$\mathbf{B} = \begin{bmatrix} 1 & 0 & -x^{22} \\ x^{17} & 1 & x \\ x^{19} & x & 1 \end{bmatrix}$$

which shares some nice properties with $\mathbf{A}$. This matrix is constructed by dividing the third column into its two x^{22}-adic coefficients (rows 2 and 3) and then including an additional row (row 1) which provides the single column operation which would undo the division. Thus by construction this matrix is unimodularly equivalent to

$$\begin{bmatrix} 1 & 0 & 0 \\ x^{17} & 1 & x^{39} + x \\ x^{19} & x & x^{41} + 1 \end{bmatrix}$$

and it is easily seen that the Hermite form of $\mathbf{A}$ will be given by the 2×2 trailing submatrix of

the Hermite form of $\mathbf{B}$. As such we rely on the computation of the Hermite form of a matrix, not much larger than the original matrix, but having the nice property that the degrees are much more uniformly distributed.

Proposition 6.1. *Let $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$ be nonsingular. Using no operation in $\mathbb{K}$, one can build a nonsingular matrix $\mathbf{B} \in \mathbb{K}[x]^{m\times m}$ such that*

(i) $n \le m < 3n$ and $\deg(\mathbf{B}) \le \lceil D(\mathbf{A})/n \rceil$,

(ii) the Hermite form of $\mathbf{A}$ is the trailing principal $n \times n$ submatrix of the Hermite form of $\mathbf{B}$.

Proof. The partial linearization used in [15, Corollary 3] takes $\mathbf{A}$ and constructs a matrix $\mathbf{C} \in \mathbb{K}[x]^{m\times m}$ with smoothed degrees having the properties: (a) $\mathbf{C}$ is a nonsingular matrix with $m < 3n$, (b) $\deg(\mathbf{C}) \le \lceil D(\mathbf{A})/n \rceil$ and (c) the principal $n \times n$ submatrix of $\mathbf{C}^{-1}$ is equal to $\mathbf{A}^{-1}$. Permuting the rows and columns of this matrix $\mathbf{C}$ into

$$\mathbf{B} \;=\; \begin{bmatrix} \mathbf{0} & \mathbf{I}_{m-n} \\ \mathbf{I}_n & \mathbf{0} \end{bmatrix} \mathbf{C} \begin{bmatrix} \mathbf{0} & \mathbf{I}_n \\ \mathbf{I}_{m-n} & \mathbf{0} \end{bmatrix} \in \mathbb{K}[x]^{m\times m},$$

we see that $\mathbf{A}^{-1}$ appears as the trailing $n \times n$ submatrix of $\mathbf{B}^{-1}$. We will prove that the Hermite form of $\mathbf{B}$ has the shape $\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ * & \mathbf{H} \end{bmatrix}$, where $\mathbf{H}$ is the Hermite form of $\mathbf{A}$.

Let $\mathbf{T} = \begin{bmatrix} \mathbf{H}_1 & \mathbf{0} \\ * & \mathbf{H}_2 \end{bmatrix}$ be the Hermite form of $\mathbf{B}$, where $\mathbf{H}_1 \in \mathbb{K}[x]^{(m-n)\times(m-n)}$ and $\mathbf{H}_2 \in \mathbb{K}[x]^{n\times n}$. We can write $\mathbf{H}_2 = \mathbf{A}\mathbf{D}$, where the matrix $\mathbf{D} = \mathbf{A}^{-1}\mathbf{H}_2$ has entries in $\mathbb{K}[x]$. Indeed, by construction,

$$\mathbf{B}^{-1}\mathbf{T} = \begin{bmatrix} * & * \\ * & \mathbf{A}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{H}_1 & \mathbf{0} \\ * & \mathbf{H}_2 \end{bmatrix} = \begin{bmatrix} * & * \\ * & \mathbf{A}^{-1}\mathbf{H}_2 \end{bmatrix}.$$

is a (unimodular) matrix in $\mathbb{K}[x]^{m\times m}$. On the other hand, according to [15, Corollary 5] we have $\det(\mathbf{B}) = \det(\mathbf{A})$, and therefore

$$\det(\mathbf{A}) = \lambda \det(\mathbf{T}) = \lambda \det(\mathbf{H}_1)\det(\mathbf{H}_2) = \lambda \det(\mathbf{H}_1)\det(\mathbf{A})\det(\mathbf{D})$$

where $\lambda = \det(\mathbf{B}^{-1}\mathbf{T})^{-1}$ is a nonzero constant from $\mathbb{K}$. Thus, $\mathbf{H}_1$ and $\mathbf{D}$ are both unimodular. Therefore, since $\mathbf{H}_1$ is in Hermite form, it must be the identity matrix and, since $\mathbf{H}_2$ is in Hermite form and right-unimodularly equivalent to $\mathbf{A}$, it must be equal to $\mathbf{H}$. □

For the details of how to build the matrix $\mathbf{C}$ using row and column partial linearization, we refer the reader to [15, Section 6]. We give here two detailed examples (see also [15, Example 4]), written with the help of our prototype implementation of the algorithms described in this paper.

Example 6.2. Let $\mathbb{K}$ be the finite field with 997 elements. Using a computer algebra system, we choose $\mathbf{A} \in \mathbb{K}[x]^{4\times 4}$ with prescribed degrees and random coefficients from $\mathbb{K}$. Instead of showing the entire matrix let us only consider the degree profile which in this case is

$$\mathbf{A} = \begin{bmatrix} [2] & [10] & [63] & [5] \\ [75] & [51] & [95] & [69] \\ [4] & [5] & [48] & [7] \\ [10] & [54] & [75] & [6] \end{bmatrix},$$

where $[d]$ indicates an entry of degree d. For the sake of presentation, we note that $D(\mathbf{A}) = 199 = 75 + 54 + 63 + 7$; however, this quantity is not computed by our algorithm. Instead, to find which degrees we will use to partially linearize the columns of $\mathbf{A}$, we permute its rows and columns to ensure that the diagonal degrees dominate the degrees in the trailing principal submatrices:

$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} [2] & [10] & [63] & [5] \\ [75] & [51] & [95] & [69] \\ [4] & [5] & [48] & [7] \\ [10] & [54] & [75] & [6] \end{bmatrix} \underbrace{\begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\boldsymbol{\pi}} = \begin{bmatrix} [95] & [51] & [69] & [75] \\ [75] & [54] & [6] & [10] \\ [48] & [5] & [7] & [4] \\ [63] & [10] & [5] & [2] \end{bmatrix} \tag{8}$$

Then, the diagonal degrees $95, 54, 7, 2$ are used for column partial linearization; we remark that $95 + 54 + 7 + 2 = 158 \leq D(\mathbf{A})$. Permuting back the rows and columns of $\mathbf{A}$, we will partially linearize its columns with respect to the degrees $(2, 54, 95, 7) = (95, 54, 7, 2)\boldsymbol{\pi}^{-1}$. Since the average of these degrees is $\lceil 158/4 \rceil = 40$, the columns are linearized into $(1, 2, 3, 1)$ columns, respectively. That is, columns 1 and 4 of $\mathbf{A}$ will not be affected, column 2 of $\mathbf{A}$ will be expanded into 2 columns, and column 3 of $\mathbf{A}$ will be expanded into 3 columns. Elementary rows are inserted at the same time to reflect these column linearizations. Thus, we obtain a column linearized version of $\mathbf{A}$ as

$$\hat{\mathbf{A}} = \begin{bmatrix} [2] & [10] & [39] & [5] & 0 & [23] & 0 \\ [75] & [39] & [39] & [69] & [11] & [39] & [15] \\ [4] & [5] & [39] & [7] & 0 & [8] & 0 \\ [10] & [39] & [39] & [6] & [14] & [35] & 0 \\ 0 & -x^{40} & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -x^{40} & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -x^{40} & 1 \end{bmatrix}.$$

In particular, we have

$$\operatorname{rdeg}(\hat{\mathbf{A}}) = (39, 75, 39, 39, 40, 40, 40),$$

whose average is $\lceil 312/7 \rceil = 45$. Now, we perform a partial linearization on the rows with respect to their row degree. Only the second row has degree $75 > 45$, and is therefore split into two rows; inserting an elementary column accordingly, we obtain

$$\mathbf{C} = \begin{bmatrix} [2] & [10] & [39] & [5] & 0 & [23] & 0 & 0 \\ [44] & [39] & [39] & [44] & [11] & [39] & [15] & -x^{45} \\ [4] & [5] & [39] & [7] & 0 & [8] & 0 & 0 \\ [10] & [39] & [39] & [6] & [14] & [35] & 0 & 0 \\ 0 & -x^{40} & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -x^{40} & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -x^{40} & 1 & 0 \\ [30] & 0 & 0 & [24] & 0 & 0 & 0 & 1 \end{bmatrix}$$

whose degree is 45. Finally, we verify that the Hermite form of $\begin{bmatrix} \mathbf{0} & \mathbf{I}_4 \\ \mathbf{I}_4 & \mathbf{0} \end{bmatrix} \mathbf{C} \begin{bmatrix} \mathbf{0} & \mathbf{I}_4 \\ \mathbf{I}_4 & \mathbf{0} \end{bmatrix}$ is $\begin{bmatrix} \mathbf{I}_4 & \mathbf{0} \\ * & \mathbf{H} \end{bmatrix}$, with $\mathbf{H}$ the Hermite form of $\mathbf{A}$. Thus, we have transformed a Hermite form problem in dimensions 4×4 and degree 95 into one in dimensions 8×8 but degree less than $50 = \lceil D(\mathbf{A})/4 \rceil$. ◇

Example 6.3. Let $\mathbb{K}$ be the field with 7 elements, and consider the matrix from Example 3.2:

$$\mathbf{A} = \begin{bmatrix} 6x+1 & 2x^3+x^2+6x+1 & 3 \\ 4x^5+5x^4+4x^2+x & 6x^5+5x^4+2x^3+4 & x^4+5x^3+6x^2+5x \\ 2 & 2x^5+5x^4+5x^3+6x^2 & 6 \end{bmatrix}.$$

Here, $D(\mathbf{A}) = \deg(\det(\mathbf{A})) = 1+5+4 = 10$. We consider a row- and column-permuted version of the matrix $\mathbf{A}$ ensuring that the diagonal degrees are dominant, as we did in Example 6.2:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{A} \underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{\pi}} = \begin{bmatrix} 6x^5+5x^4+2x^3+4 & 4x^5+5x^4+4x^2+x & x^4+5x^3+6x^2+5x \\ 2x^3+x^2+6x+1 & 6x+1 & 3 \\ 2x^5+5x^4+5x^3+6x^2 & 2 & 6 \end{bmatrix}.$$

This gives us the linearization degrees $(1,5,0) = (5,1,0)\boldsymbol{\pi}^{-1}$, which have average $\lceil 6/3 \rceil = 2$, so the partial column linearization results in

$$\hat{\mathbf{A}} = \begin{bmatrix} 6x+1 & 6x+1 & 3 & 2x+1 & 0 \\ 4x^5+5x^4+4x^2+x & 4 & x^4+5x^3+6x^2+5x & 2x & 6x+5 \\ 2 & 0 & 6 & 5x+6 & 2x+5 \\ 0 & 6x^2 & 0 & 1 & 0 \\ 0 & 0 & 0 & 6x^2 & 1 \end{bmatrix}.$$

Then, we perform row partial linearization of this matrix with respect to its row degrees $(1,5,1,2,2)$, whose average is $\lceil 11/5 \rceil = 3$, giving

$$\mathbf{C} = \begin{bmatrix} 6x+1 & 6x+1 & 3 & 2x+1 & 0 & 0 \\ 4x^2+x & 4 & 6x^2+5x & 2x & 6x+5 & 6x^3 \\ 2 & 0 & 6 & 5x+6 & 2x+5 & 0 \\ 0 & 6x^2 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 6x^2 & 1 & 0 \\ 4x^2+5x & 0 & x+5 & 0 & 0 & 1 \end{bmatrix}.$$

Using the algorithm in Section 3, we obtain the degrees $(0,0,0,0,1,9)$ of the diagonal entries of the Hermite form of the permuted matrix

$$\mathbf{B} = \begin{bmatrix} \mathbf{0} & \mathbf{I}_3 \\ \mathbf{I}_3 & \mathbf{0} \end{bmatrix} \mathbf{C} \begin{bmatrix} \mathbf{0} & \mathbf{I}_3 \\ \mathbf{I}_3 & \mathbf{0} \end{bmatrix}.$$

Proceeding then as in Section 5, we can to compute the complete Hermite form of $\mathbf{B}$ using the knowledge of these degrees, giving

$$\begin{bmatrix} \mathbf{I}_3 & \mathbf{0} \\ \mathbf{R} & \mathbf{H} \end{bmatrix}$$

where $\mathbf{H}$ is the Hermite form of $\mathbf{A}$ as given in Example 5.10, and the transpose of $\mathbf{R}$ is

$$\mathbf{R}^{\mathsf{T}} = \begin{bmatrix} 0 & 6 & 4x^7+6x^6+x^5+4x^4+2x^3+6x^2+4 \\ 0 & 3 & 6x^8+4x^7+4x^5+3x^4+3x^3+2x+6 \\ 0 & 4 & 3x^8+2x^7+3x^6+3x^5+4x^3+5x^2+x+2 \end{bmatrix}. \quad \diamond$$

7. Conclusion

In this paper we have given new, deterministic algorithms for computing the Hermite normal form and the determinant of a nonsingular polynomial matrix. Our methods are based on the efficient, deterministic computation of the diagonal elements of the Hermite form. While our algorithms are fast in terms of the number of operations in an abstract field $\mathbb{K}$, they do not take into consideration the possible growth of coefficients in the field, an issue when working over certain fields such as $\mathbb{Q}$, the rational numbers. Kannan [24] was the first to show that computing Hermite normal forms over $\mathbb{Q}[x]$ can be done in polynomial time. Fraction-free algorithms for Hermite form computation which take into consideration coefficient growth have been given in [8] (using a shifted Popov algorithm) and [25] (where the problem is converted into a large linear system). We plan to investigate exact algorithms for Hermite and determinant computation based on the fraction-free approach used in [6] and also the use of Chinese remaindering. In the latter case the reduced domains (e.g. $\mathbb{Z}_p[x]$) do not encounter coefficient growth which allows for effective use of the algorithms in this paper. The issue in this case is the reconstruction of the images, where we expect the techniques used in [10] will be helpful.

In terms of additional future research we are interested in the still open problem of reducing computation of the Hermite form over the integers [33] to the complexity of integer matrix multiplication. In addition, we are interested in finding efficient, deterministic algorithms for other normal forms of polynomial matrices, such as the Popov normal form, or more generally the shifted Popov normal forms. In addition we are interested in fast normal form algorithms where the entries are differential operators rather than polynomials. Such algorithms are useful for reducing systems of linear differential equations to solvable first order systems [3].

Acknowledgments. The authors would like to thank Arne Storjohann and an anonymous referee for suggestions on simplifying the presentation of Section 6 and about the alternative determinant algorithm in the Appendix. G. Labahn was supported by a grant from NSERC while V. Neiger was supported by the international mobility grants from *Projet Avenir Lyon Saint-Étienne*, *Mitacs Globalink - Inria*, and Explo'ra Doc from *Région Rhône-Alpes*.

References

[1] J. Abbott, M. Bronstein, and T. Mulders. Fast deterministic computation of determinants of dense matrices. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC'99, pages 197–204. ACM Press, 1999.

[2] M. Van Barel and A. Bultheel. A general module theoretic framework for vector M-Padé and matrix rational interpolation. Numerical Algorithms, 3:451–462, 1992.

[3] M. Barkatou, C. El Bacha, G. Labahn, and E. Pflügel. On simultaneous row and column reduction of higher-order linear differential systems. *Journal of Symbolic Computation*, 49(1):45–64, 2013.

[4] B. Beckermann. A reliable method for computing M-Padé approximants on arbitrary staircases. Journal of Computational and Applied Mathematics, 40:19–42, 1992.

[5] 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.

[6] 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.

[7] B. Beckermann, G. Labahn, and G. Villard. Shifted normal forms of polynomial matrices. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC'99, pages 189–196, 1999.

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

[9] J. Bunch and J. Hopcroft. Triangular factorization and inversion by fast matrix multiplication. *Mathematics of Computation*, 28:231–236, 1974.

[10] H. Cheng and G. Labahn, Modular Computation for Matrices of Ore Polynomials, In *Proceedings of WSPC (In Honor of the 60-th birthday of Sergei Abramov)*, (2007) 43-66

[11] D. S. Dummit and R. M. Foote. *Abstract Algebra*. John Wiley & Sons, 2004.

[12] W. Eberly, M. Giesbrecht, and G. Villard. On computing the determinant and Smith normal form of an integer matrix. In *Proceedings of 41st IEEE Symposium on Foundations of Computer Science (FOCS'2000)*, pages 675–687, 2000.

[13] P. Giorgi, C.-P. Jeannerod, and G. Villard. On the complexity of polynomial matrix computations. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation, Philadelphia, Pennsylvania, USA*, ISSAC'03, pages 135–142. ACM Press, 2003.

[14] S. Gupta. Hermite forms of polynomial matrices. Master's thesis, University of Waterloo, 2011.

[15] 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.

[16] S. Gupta and A. Storjohann. Computing Hermite forms of polynomial matrices. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC'11, pages 155–162, 2011.

[17] J. Hafner and K. McCurley. Asyptotically fast triangularization of matrices over rings. *SIAM Journal of Computing*, 20:1068–1083, 1991.

[18] C. Hermite. Sur l'introduction des variables continues dans la théorie des nombres. *Journal für die reine und angewandte Mathematik*, 41:191–216, 1851.

[19] C. Iliopoulos. Worst-case complexity bounds on algorithms for computing the canonical structure of finite abelian groups and the Hermite and Smith normal forms of integer matrices. *SIAM Journal of Computing*, 18:658–669, 1986.

[20] C.-P. Jeannerod, V. Neiger, E. Schost and G. Villard. Fast computation of minimal interpolation bases in Popov form for arbitrary shifts. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC'16, pages 295–302. ACM, 2016.

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

[22] E. Kaltofen. On computing determinants of matrices without divisions. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC'92, pages 342–349. ACM, 1992.

[23] E. Kaltofen and G. Villard. On the complexity of computing determinants. *Computational Complexity*, 13:91–130, 2004.

[24] R. Kannan. Polynomial-time algorithms for solving systems of linear equations over polynomials. *Theoretical Computer Science*, 39:69–88, 1985.

[25] S.E. Labhalla, H. Lombardi and R. Marlin. Algorithmes de calcul de la réduction d'Hermite d'une matrice à coefficients polynmiaux. In *Comptes-Rendus de MEGA92*, Nice, France, 1992.

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

[27] V. Neiger. Fast computation of shifted Popov forms of polynomial matrices via systems of modular polynomial equations. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC'16, pages 365–372. ACM, 2016.

[28] S. Sarkar and A. Storjohann. Normalization of row reduced matrices. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC'11, pages 297–304, 2011.

[29] A. Storjohann. *Algorithms for Matrix Canonical Forms*. PhD thesis, Department of Computer Science, Swiss Federal Institute of Technology—ETH, 2000.

[30] A. Storjohann. High-order lifting and integrality certification. *Journal of Symbolic Computation,*, 36:613–648, 2003.

[31] A. Storjohann. Notes on computing minimal approximant bases. In *Dagstuhl Seminar Proceedings*, ISSN:1862-4405, 2006.

[32] A. Storjohann. On the complexity of inverting integer and polynomial matrices. A. comput. complex. (2015) 24:777.

[33] A. Storjohann and G. Labahn. Asymptotically fast computation of Hermite forms of integer matrices. In *International Symposium on Symbolic and Algebraic Computation*, ISSAC'96, pages 259–266, 1996.

[34] A. Storjohann and G. Villard. Computing the rank and a small nullspace basis of a polynomial matrix. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC'05, pages 309–316, 2005.

[35] G. Villard. Computing Popov and Hermite forms of polynomial matrices. In *International Symposium on Symbolic and Algebraic Computation*, ISSAC'96, pages 250–258, 1996.

[36] W. Zhou. *Fast order basis and kernel basis computation and related problems*. PhD thesis, University of Waterloo, 2012.

[37] W. Zhou and G. Labahn. Efficient computation of order bases. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC'09, pages 375–382. ACM, 2009.

[38] W. Zhou and G. Labahn. Efficient algorithms for order basis computation. *Journal of Symbolic Computation*, 47(7):793–819, 2012.

[39] W. Zhou and G. Labahn. Computing column bases of polynomial matrices. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC'13, pages 379–387. ACM, 2013.

[40] W. Zhou and G. Labahn. Unimodular completion of polynomial matrices. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC'14, pages 413–420. ACM, 2014.

[41] W. Zhou, G. Labahn, and A. Storjohann. Computing minimal nullspace bases. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC'12, pages 375–382. ACM, 2012.

[42] W. Zhou, G. Labahn and A. Storjohann. A deterministic algorithm for inverting a polynomial matrix. *Journal of Complexity*, 31(2):162–173, 2015.

Appendix. Another fast and deterministic algorithm for the determinant

In this appendix, we describe an alternative to our determinant Algorithm 2, kindly suggested by a reviewer. The main idea is to rely on x-Smith decomposition [15] in order to make sure that the determinant can be easily retrieved from the diagonal entries of a triangular form computed with Algorithm 1. This is thus a way to overcome the obstacle mentioned in Remark 4.1.

Let $\mathbf{A} \in \mathbb{K}[x]^{n\times n}$ be a nonsingular polynomial matrix. Then, [15, Corollary 1] states that we can compute a triangular x-Smith decomposition of $\mathbf{A}$ using $\widetilde{\mathcal{O}}(n^\omega \deg(\mathbf{A}))$ field operations. This yields matrices $\pi, \mathbf{U}, \mathbf{H}$ such that $\mathbf{A}\pi = \mathbf{U}\mathbf{H}$, where

- $\pi \in \mathbb{K}^{n\times n}$ is a permutation matrix,
- $\mathbf{H} \in \mathbb{K}[x]^{n\times n}$ is triangular with $\det(\mathbf{H}) = x^\alpha$ for some $\alpha \in \mathbb{N}$,
- $\mathbf{U} \in \mathbb{K}[x]^{n\times n}$ is such that $\det(\mathbf{U} \bmod x) \neq 0$ and $\deg(\mathbf{U}) \leq \deg(\mathbf{A})$.

Then, we have $\det(\mathbf{A}) = \det(\mathbf{U})\det(\mathbf{H})\det(\pi)^{-1}$, and the cost of finding $\det(\mathbf{H})$ and $\det(\pi)$ is negligible. It remains to compute $\det(\mathbf{U})$, which can be done in $\widetilde{\mathcal{O}}(n^\omega \deg(\mathbf{U})) \subseteq \widetilde{\mathcal{O}}(n^\omega \deg(\mathbf{A}))$ operations. Indeed, since $\det(\mathbf{U} \bmod x) \neq 0$, determining the diagonal entries of a triangular form of $\mathbf{U}$ allows us to deduce its determinant as explained in Remark 4.1.

Thus, we obtain $\det(\mathbf{A})$ in $\widetilde{\mathcal{O}}(n^\omega \deg(\mathbf{A}))$ field operations; with Proposition 4.6, this gives another proof of Theorem 1.1.