On lattice reduction for polynomial matrices

Technical Report No. 356, Department of Computer Science, ETH Zurich, December 2000

On Lattice Reduction for Polynomial Matrices

T. Mulders and A. Storjohann

Institute of Scientific Computing
ETH Zurich, Switzerland
{mulders, storjoha}*@inf.ethz.ch*

Abstract

A simple algorithm for transformation to weak Popov form — essentially lattice reduction for polynomial matrices — is described and analyzed. The algorithm is adapted and applied to various tasks involving polynomial matrices: rank profile and determinant computation; unimodular triangular factorization; transformation to Hermite and Popov canonical form; rational and diophantine linear system solving; short vector computation.

1. Introduction

Let A be a matrix over $F[x]$, F a field. By applying a sequence of elementary row operations we can transform A to a matrix R which is in weak Popov form. An example is given in Figure 1. We defer until Section 2 to define the form

$$\begin{bmatrix} 4x^2+3x+5 & 4x^2+3x+4 & 6x^2+1 \\ 3x+6 & 3x+5 & 3+x \\ 6x^2+4x+2 & 6x^2 & 2x^2+x \end{bmatrix} \longrightarrow \begin{bmatrix} 1 & 6x+3 & 6 \\ 0 & 0 & 0 \\ 2 & 5 & 3 \end{bmatrix}$$

Figure 1: Transformation of a 3×3 rank 2 matrix over $\mathbb{Z}/(7)$ to weak Popov form.

precisely. For now, we note two key properties of the weak Popov form:

- R will have rank A nonzero rows, and
- the sum of the degrees of the nonzero rows is minimal among all matrices which can be obtained from A by applying elementary row transformations.

Thus, transformation to weak Popov form is essentially lattice reduction for polynomial matrices. The weak Popov form is a simplified, non–canonical version of the well known Popov canonical form from linear control theory.

This paper gives a simple algorithm to transform a polynomial matrix $A \in F[x]^{n\times m}$ to weak Popov form. The cost of the algorithm is $O(nmrd^2)$ field operations where r is the rank of A. We adapt and apply the algorithm to get solutions to various other problems involving A, including transformation to canonical Popov form. Table 1 lists all the problems we consider and also gives the organization of the bulk of this paper. All the problems listed are also solved in $O(nmrd^2)$ field operations, except for diophantine linear system solving, which is solved in $O((n+m)^3d^2)$ field operations. By diophantine linear system solving we mean to express, if possible, the first row of A as an $F[x]$-linear combination of the remaining rows. By triangular factorization we mean to express A as UH where H is upper triangular (eg. in Hermite form) and U is unimodular. Many of the latter sections listed in Table 1 depend on previous sections.

§2 Transformation to weak Popov form and recovery of rank profile.

§3 Computation of determinant.

§4 Transformation of full column rank input matrix to Hermite form.

§5 Unimodular triangular factorization of a full column rank input matrix.

§6 Diophantine linear system solving.

§7 Transformation to Popov form.

§8 Transformation to reduced basis and computation of short vectors.

Table 1: Some polynomial matrix computations.

An algorithm to compute a reduced basis very similar to the weak Popov form has been given by von zur Gathen (1984) and applied to the problem of computing short vectors. In Section 8 we indicate the relationship between Popov form and *reduced basis* as defined in von zur Gathen (1984). This results in a substantially faster algorithm for the reduced bases and short vectors problem.

In Section 9 we extend the notion of weak Popov form to the setting of discrete valuation rings. Analogous results as in the polynomial setting hold.

We end the paper with a short summary, some remarks on implementation issues and some suggestions for further research.

This paper is based on the poster Mulders and Storjohann (2000b).

2. The weak Popov form

A well–known notion in systems theory is the Popov form of a rectangular matrix with polynomial entries. A weaker but also useful form is the quasi–Popov form (Kailath (1980)). In this section we introduce an even weaker form — the weak–Popov form. In essence, the Popov form is a weak–Popov form with sundry other conditions on the rows. Since these additional conditions will not be useful for

our applications of the form in subsequent sections, we restrict ourselves to the simpler weak Popov form.

Let F be a field and $M = (m_{i,j}) \in F[x]^{n \times m}$. In what follows we use M to define general notions for matrices. We use calligraphic characters to refer to specific variables used in the various algorithms.

Definition: For $1 \leq i \leq n$ we define the *ith pivot index* I_i^M of M as follows: if $m_{i,j} = 0$ for $1 \leq j \leq m$, then $I_i^M = 0$; otherwise

1. $\deg(m_{i,j}) \leq \deg(m_{i,I_i^M})$ for $1 \leq j < I_i^M$;
2. $\deg(m_{i,j}) < \deg(m_{i,I_i^M})$ for $I_i^M < j \leq m$.

When $I_i^M \neq 0$, the element m_{i,I_i^M} is called the *ith pivot element* of M and is denoted by P_i^M. The degree of P_i^M is called the *ith pivot degree* of M and is denoted by D_i^M. When $I_i^M = 0$ we put $D_i^M = -1$.

A pivot element is the rightmost element with maximum degree in its row. Note that in particular $\deg(m_{i,j}) \leq D_i^M$ for all $1 \leq j \leq m$. In what follows, this fact will be used without further notice.

Definition: The *carrier set* C^M of M is defined as $C^M = \{1 \leq i \leq n \mid I_i^M \neq 0\}$.

Definition: M is said to be in weak Popov form if the positive pivot indices of M are all different, i.e. if

$$k, l \in C^M, k \neq l \quad \Rightarrow \quad I_k^M \neq I_l^M.$$

By applying unimodular row transformations, we want to transform a given matrix into weak Popov form. We now define a particularly simple kind of unimodular transformation.

Definition: If $k \in C^M$, $l \neq k$ and $\deg(m_{l,I_k^M}) \geq D_k^M$, there are unique $c \in F$ and $e \in \mathbb{N}$ such that

$$\deg(m_{l,I_k^M} - cx^e P_k^M) < \deg(m_{l,I_k^M}).$$

In that case we call subtracting cx^e times row k from row l the *simple transformation* of row k on row l. If $I_l^M = I_k^M$, the transformation is called of the *first kind*, otherwise it is called of the *second kind*.

Sometimes we want to apply a simple transformation on M and meanwhile apply the same transformation on a vector or matrix A. We then say that we apply the transformation on $\left[\begin{array}{c|c} M & A \end{array}\right]$. Notice that in that case we only consider M when we determine the pivot element of a row.

Definition: When $\left[\begin{array}{c|c} N & B \end{array}\right]$ is the result after applying a number of simple transformations on $\left[\begin{array}{c|c} M & A \end{array}\right]$, we write $\left[\begin{array}{c|c} M & A \end{array}\right] \rightarrow \left[\begin{array}{c|c} N & B \end{array}\right]$. Note that in that case $\left[\begin{array}{c|c} N & B \end{array}\right]$ is left equivalent to $\left[\begin{array}{c|c} M & A \end{array}\right]$, i.e. $\left[\begin{array}{c|c} N & B \end{array}\right] = U \left[\begin{array}{c|c} M & A \end{array}\right]$ where U is unimodular and even $\det(U) = 1$.

algorithm WeakPopovForm
input: $\mathcal{M} \in F[x]^{n \times m}$.
output: $\mathcal{N}$ in weak Popov form, obtained by applying simple transformations of the first kind on $\mathcal{M}$.
$\mathcal{A} := \text{copy}(\mathcal{M})$;
while $\mathcal{A}$ is not in weak Popov form **do**
    Apply a simple transformation of the first kind on $\mathcal{A}$
od;
$\mathcal{N} := \text{copy}(\mathcal{A})$;
return $\mathcal{N}$

Figure 2: Algorithm *WeakPopovForm*

EXAMPLE 1: *Let*

$$M = \begin{bmatrix} 1 & x^2 & x \\ 3x & x + 2x^3 & x^3 \end{bmatrix}, \quad A = \begin{bmatrix} x^4 \\ x^2 \end{bmatrix}$$

and

$$N = \begin{bmatrix} 1 & x^2 & x \\ x & x & x^3 - 2x^2 \end{bmatrix}, \quad B = \begin{bmatrix} x^4 \\ x^2 - 2x^5 \end{bmatrix}.$$

Then $I_1^M = 2$ *and by applying the simple transformation of the first row on the second row of* $\left[\begin{array}{c|c} M & A \end{array}\right]$, *we see that* $\left[\begin{array}{c|c} M & A \end{array}\right] \rightarrow \left[\begin{array}{c|c} N & B \end{array}\right]$.

The following trivial lemma will lead to an algorithm to transform a matrix into weak Popov form applying only simple transformations of the first kind.

LEMMA 2.1: *M is not in weak Popov form if and only if we can apply a simple transformation of the first kind on M, that is, not all non–zero pivot indices of M are different.*

Figure 2 describes a simple algorithm to transform a polynomial matrix into weak Popov form. The correctness of the algorithms outcome is evident from Lemma 2.1. We have to show however that the algorithm will eventually stop. Notice that the copying of matrices in Algorithm *WeakPopovForm* is not necessary. We only do this in order to be able to reason about the algorithm.

The following lemma expresses how the pivot indices and pivot degrees may change when we apply a simple transformation.

LEMMA 2.2: *Let N be the matrix we get after applying the simple transformation of row k on row l of M. Then*

$$I_i^N = I_i^M, D_i^N = D_i^M \text{ for } i \neq l; \tag{1}$$

$$D_l^N \leq D_l^M. \tag{2}$$

If the simple transformation is of the first kind, then either $D_l^N < D_l^M$ or ($D_l^N = D_l^M$ and $I_l^N < I_l^M$). If the simple transformation is of the second kind, then $I_l^N = I_l^M$ and $D_l^N = D_l^M$.

Proof: (1) is clear, since only row l changes. Write $N = (n_{i,j})$. From the definition of simple transformation it follows that for $1 \leq j \leq m$ we have

$$\deg(n_{l,j}) \leq \max(\deg(m_{l,j}), \deg(m_{k,j}) + \deg(m_{l,I_k^M}) - D_k^M), \tag{3}$$

with equality if $\deg(m_{l,j}) \neq \deg(m_{k,j}) + \deg(m_{l,I_k^M}) - D_k^M$.

Since $\deg(m_{l,j}), \deg(m_{l,I_k^M}) \leq D_l^M$ and $\deg(m_{k,j}) \leq D_k^M$ we see from (3) that $\deg(n_{l,j}) \leq D_l^M$ and (2) follows.

Now suppose that the simple transformation is of the first kind, i.e. $I_l^M = I_k^M$. Then the definition of simple transformation implies that $\deg(n_{l,I_l^M}) < \deg(m_{l,I_l^M}) = D_l^M$. For $I_k^M = I_l^M < j \leq m$ we have $\deg(m_{l,j}) < D_l^M$ and $\deg(m_{k,j}) < D_k^M$ and we see from (3) that $\deg(n_{l,j}) < D_l^M$. So if $D_l^N = D_l^M$, then $I_l^N < I_l^M$.

Finally suppose that the simple transformation is of the second kind, i.e $I_l^M \neq I_k^M$. Then either $\deg(m_{k,I_l^M}) < D_k^M$ or $\deg(m_{l,I_k^M}) < D_l^M$, and it follows from (3) that $\deg(n_{l,I_l^M}) = \deg(m_{l,I_l^M}) = D_l^M$. From (2) we see that $\deg(n_{l,j}) \leq D_l^M$ for $1 \leq j < I_l^M$. For $I_l^M < j \leq m$ we have $\deg(m_{l,j}) < D_l^M$ and if $\deg(m_{k,j}) = D_k^M$, then $I_l^M < j \leq I_k^M$, implying that $\deg(m_{l,I_k^M}) < D_l^M$. From (3) we now see that $\deg(n_{l,j}) < D_l^M$ for $I_l^M < j \leq m$. It follows that $I_l^N = I_l^M$. □

THEOREM 2.1: *Algorithm* WeakPopovForm *is correct.*

Proof: Since $I_i^{\mathcal{A}} \leq m$ for $1 \leq i \leq n$, it follows from Lemma 2.2 that $\sum_{i \in C^{\mathcal{A}}} (mD_i^{\mathcal{A}} + I_i^{\mathcal{A}}) \geq 0$ decreases every time when a simple transformation of the first kind is applied on $\mathcal{A}$. This shows that the algorithm will eventually stop. The theorem now follows from Lemma 2.1. □

Now we bound the cost of Algorithm *WeakPopovForm*. For this the following is important to notice.

COROLLARY 2.1: *When d is a bound on the degrees of the entries of $\mathcal{M}$, then the degrees of all entries of $\mathcal{A}$ are always bounded by d.*

Proof: This follows immediately from Lemma 2.2 □

Now we describe the possible values that a pair $(D_l^{\mathcal{A}}, I_l^{\mathcal{A}})$ can assume during the course of Algorithm *WeakPopovForm*.

Definition: The set $I^M = \{I_i^M \mid i \in C^M\}$ of non–zero pivot indices of M is called the *index set* of M.

LEMMA 2.3: *When N is the matrix we get after applying a simple transformation on M, then $I^M \subseteq I^N$.*

Proof: When we apply the simple transformation of row k on row l, we see from Lemma 2.2 that only I_l may change, when the simple transformation is of the first kind. In that case $I_l^M = I_k^M \in I^N$. The lemma follows. □

LEMMA 2.4: *For $1 \leq l \leq n$, the values that the pair $(D_l^{\mathcal{A}}, I_l^{\mathcal{A}})$ can assume during the course of Algorithm* WeakPopovForm *are all in the set $\{D_l^{\mathcal{N}}, D_l^{\mathcal{N}} + 1, \ldots, D_l^{\mathcal{M}}\} \times (I^{\mathcal{N}} \cup \{0\})$.*

Proof: That the values of $I_l^{\mathcal{A}}$ are in $I_{\mathcal{N}} \cup \{0\}$ follows from Lemma 2.3. That the values of $D_l^{\mathcal{A}}$ are in $\{D_l^{\mathcal{N}}, D_l^{\mathcal{N}} + 1, \ldots, D_l^{\mathcal{M}}\}$ follows from Lemma 2.2. □

LEMMA 2.5: *If the pivot indices of all rows of M are positive and different, then the rows of M are independent over $F(x)$.*

Proof: Let N be the matrix we get by multiplying, for $1 \leq i \leq n$, row i by $x^{-D_i^M}$. Then $N = N_0 + \hat{N}$, where $\hat{N} \in x^{-1}F[x^{-1}]^{n \times m}$ and $N_0 \in F^{n \times m}$ has independent rows. Consider $F(x) \subset F((x^{-1}))$. It is clear that the rows of N are independent over $F((x^{-1}))$ and thus are also independent over $F(x)$. □

COROLLARY 2.2: $\text{rank}(M) \geq \#I^M$.

Proof: Apply Lemma 2.5 on $\#I^M$ rows of M whose indices constitute the set I^M. □

THEOREM 2.2: *When d is a bound on the degrees of the entries of $\mathcal{M}$ and r is the rank of $\mathcal{M}$, then the cost of Algorithm* WeakPopovForm *is bounded by $O(nmrd^2)$ field operations.*

Proof: From Lemma 2.4 it follows that, during the course of the algorithm, the pair $(D_l^{\mathcal{A}}, I_l^{\mathcal{A}})$ can assume at most $(D_l^{\mathcal{M}} + 2)(\#I^{\mathcal{N}} + 1)$ values. Since $\text{rank}(\mathcal{N}) = \text{rank}(\mathcal{M})$, it follows from Corollary 2.2 that $\#I^{\mathcal{N}} = r$. By Lemma 2.2, every simple transformation of the first kind decreases for one l the pair $(D_l^{\mathcal{A}}, I_l^{\mathcal{A}})$ in the lexicographic order. It follows that the number of simple transformations applied during the course of the algorithm is $O(nrd)$. From Corollary 2.1 it follows that the cost of one simple transformation is bounded by $O(md)$. The theorem follows. □

Of course we can also determine the rank of a polynomial matrix by computing its weak Popov form and counting the number of non–zero rows.

COROLLARY 2.3: *When d is a bound on the degrees of the entries of M and r is the rank of M, then the cost of computing the rank of M is bounded by $O(nmrd^2)$ field operations.*

To be able to compute the amortized cost of some algorithms we have to specify in more detail the number of simple transformations applied by Algorithm *WeakPopovForm*.

Definition: The *state* S^M of M is defined by

$$S^M = \sum_{i \in C^M} (D_i^M m + I_i^M).$$

LEMMA 2.6: *$S^M \geq 0$. Moreover, when N is the matrix we get after applying a simple transformation of the first kind on M, then $S^N < S^M$.*

Proof: $S^M \geq 0$ follows from the fact that $D_i^M \geq 0, I_i^M \geq 1$ for all $i \in C^M$. From Lemma 2.2, the fact that $I_i^M \geq 1$ for $i \in C^M$ and $I_i^N \leq m$ for $i \in C^N$, it follows that $S^N < S^M$. □

So the state of M is a bound on the number of simple transformations of the first kind it will take to transform M into weak Popov form.

Definition: If $M \to N$, the *state drop* $S^{M,N}$ from M to N is defined by $S^{M,N} = S^M - S^N$.

THEOREM 2.3: *The number of simple transformations applied by Algorithm* WeakPopovForm *is at most $S^{\mathcal{M},\mathcal{N}}$.*

Proof: When the algorithm applies a simple transformation of the first kind, $S^{\mathcal{A}}$ decreases by Lemma 2.6. Since the first value of $S^{\mathcal{A}}$ is $S^{\mathcal{M}}$ and the last value of $S^{\mathcal{A}}$ is $S^{\mathcal{N}}$, at most $S^{\mathcal{M}} - S^{\mathcal{N}}$ simple transformations are applied. □

In fact S^M can also be defined with m replaced by $r = \text{rank}(M)$ and Theorem 2.3 then still holds. Since the proof is more involved and we do not need this result in what follows, we restrict ourselves to the current definition.

3. The determinant

In this section we show how the algorithm to compute the weak Popov form of a matrix can be used to compute the determinant of a polynomial square matrix. For this we have to adjust Algorithm *WeakPopovForm* a little. In Figure 3 the weak Popov form algorithm applies simple transformations on $\mathcal{M}$ to obtain the weak Popov form $\mathcal{N}$ and meanwhile applies the same transformations on the vector $\mathcal{V}$, obtaining $\mathcal{W}$.

To compute the cost of Algorithm *ExtendedWeakPopovForm* we have to bound the degree of the entries in $\mathcal{U}$.

Definition: The *degree sum* D^M of M is defined by

$$D^M = \sum_{i=1}^{n} D_i^M.$$

LEMMA 3.1: *When N is the matrix we get after applying a simple transformation on M, then $D^N \leq D^M$.*

algorithm ExtendedWeakPopovForm
input: $\mathcal{M} \in F[x]^{n\times m}, \mathcal{V} \in F[x]^n$.
output: $\left[\begin{array}{c|c} \mathcal{N} & \mathcal{W} \end{array}\right]$ with $\mathcal{N}$ in weak Popov form, obtained by applying simple transformations
of the first kind on $\left[\begin{array}{c|c} \mathcal{M} & \mathcal{V} \end{array}\right]$.
$(\mathcal{A},\mathcal{U}) := \text{copy}(\mathcal{M},\mathcal{V})$;
while $\mathcal{A}$ is not in weak Popov form **do**
Apply a simple transformation of the first kind on $\left[\begin{array}{c|c} \mathcal{A} & \mathcal{U} \end{array}\right]$
od;
$(\mathcal{N},\mathcal{W}) := \text{copy}(\mathcal{A},\mathcal{U})$;
return $\left[\begin{array}{cc} \mathcal{N} & \mathcal{W} \end{array}\right]$

Figure 3: Algorithm *ExtendedWeakPopovForm*

Proof: This follows immediately from Lemma 2.2. □

Definition: If $M \to N$, the *degree drop* $D^{M,N}$ from M to N is defined by $D^{M,N} = D^M - D^N$.

LEMMA 3.2: *Let* $v \in F[x]^n$ *and assume that* $\left[\begin{array}{c|c} M & v \end{array}\right] \to \left[\begin{array}{c|c} N & w \end{array}\right]$. *If* $c \in \mathbb{Z}$ *is such that* $\deg(v_i) \le D_i^M + c$ *for all* i, *then* $\deg(w_i) \le D_i^N + c + D^{M,N}$ *for all* i.

Proof: Since degree drop is additive, we only have to prove the lemma when applying one simple transformation. Suppose we apply the simple transformation of row k on row l. For $i \neq l$ we have $\deg(w_i) = \deg(v_i) \le D_i^M + c = D_i^N + c \le D_i^N + c + D^{M,N}$, since $D^{M,N} \ge 0$ by Lemma 3.1. Let $j = I_k^M$ and $M = (m_{i,j})$. Then

$$\begin{aligned} \deg(w_l) &\le \max(\deg(v_l), \deg(m_{l,j}) - \deg(m_{k,j}) + \deg(v_k)) \\ &\le \max(D_l^M + c, D_l^M - D_k^M + D_k^M + c) \\ &= D_l^M + c. \end{aligned}$$

Since $D^{M,N} = D_l^M - D_l^N$ we have $D_l^M + c = D_l^N + c + D^{M,N}$. The lemma follows. □

THEOREM 3.1: *If* d *is a bound on the degrees of the entries of* $\mathcal{M}$ *and* $\mathcal{V}$, *then the cost of Algorithm* ExtendedWeakPopovForm *is bounded by* $O((m+n)dS^{\mathcal{M},\mathcal{N}})$ *field operations.*

Proof: By Theorem 2.3 at most $S^{\mathcal{M},\mathcal{N}}$ simple transformations are applied. By Corollary 2.1 the degrees of the entries in $\mathcal{A}$ are always bounded by d. Since $\deg(\mathcal{V}_i) \le D_i^{\mathcal{M}} + d + 1$ for all i and always $D^{\mathcal{M},\mathcal{A}} \le n(d+1)$, it follows from Lemma 3.2 that the degrees of the entries in $\mathcal{U}$ are always bounded by $d + (d + 1) + n(d+1) = O(nd)$. From this the theorem follows. □

algorithm Determinant
input: $\mathcal{T} \in F[x]^{n\times n}$.
output: $\det(\mathcal{T})$.
$\bar{\mathcal{T}} := \text{copy}(\mathcal{T})$;
$\det := 1$;
for i **from** $n-1$ **by** -1 **to** 1 **do**
  $\mathcal{M} :=$ first i columns of $\bar{\mathcal{T}}$;
  $\mathcal{V} :=$ last column of $\bar{\mathcal{T}}$;
  $\left[\begin{array}{c|c} \mathcal{N} & \mathcal{W} \end{array}\right] := \text{ExtendedWeakPopovForm}(\mathcal{M}, \mathcal{V})$;
  Let k such that kth row of $\mathcal{N}$ is zero;
  $\bar{\mathcal{T}} := \mathcal{N}$ with row k deleted;
  $t :=$ kth entry of $\mathcal{W}$;
  $\det := (-1)^{k+i+1}\, t\, \det$
od;
return $\bar{\mathcal{T}}_{1,1}\, \det$

Figure 4: Algorithm *Determinant*

Let $T \in F[x]^{n\times n}$. Write $T = \left[\begin{array}{c|c} M & V \end{array}\right]$, where M consists of the first $n-1$ rows of T and V is the last column of T. Apply Algorithm *ExtendedWeakPopovForm* on the pair (M, V) yielding $\left[\begin{array}{c|c} N & W \end{array}\right]$. Since N is in weak Popov form and $\text{rank}(N) = \text{rank}(M) \le n-1$, it follows from Corollary 2.2 that N will contain at least one zero–row. So up to a row permutation we have

$$\left[\begin{array}{c|c} N & W \end{array}\right] = \left[\begin{array}{c|c} \bar{T} & * \\ 0 & t \end{array}\right],$$

where $\bar{T} \in F[x]^{(n-1)\times(n-1)}$ and $t \in F[x]$. Thus, up to sign we have

$$\det(T) = \det(\left[\begin{array}{c|c} N & W \end{array}\right]) = \det(\bar{T})\, t.$$

This leads to the algorithm to compute the determinant of a polynomial matrix described in Figure 4. Figure 5 is (up to row permutation) a pictorial representation of the flow of Algorithm *Determinant*. Here, the dark gray areas represent $\mathcal{M}$ and $\mathcal{N}$, the middle gray areas represent $\mathcal{V}$ and $\mathcal{W}$, the light gray areas are ignored during the computation and the white areas represent zero entries. The determinant of the matrix is (up to sign) the product of the black entries.

THEOREM 3.2: *When d is a bound on the degrees of the entries of $\mathcal{T}$, then the cost of Algorithm* Determinant *is bounded $O(n^3 d^2)$ field operations.*

Proof: By Corollary 2.1 the degrees of the entries of $\bar{\mathcal{T}}, \mathcal{M}, \mathcal{V}$ and $\mathcal{N}$ are always bounded by d. Let $\mathcal{M}_{n-1}, \mathcal{M}_{n-2}, \ldots, \mathcal{M}_1$ be the consecutive values of $\mathcal{M}$ and $\mathcal{N}_{n-1}, \mathcal{N}_{n-2}, \ldots, \mathcal{N}_1$ the consecutive values of $\mathcal{N}$ during the course of the algorithm. By Theorem 3.1 the cost is then bounded by

$$O\left(nd \sum_{i=1}^{n-1} S^{\mathcal{M}_i, \mathcal{N}_i}\right).$$

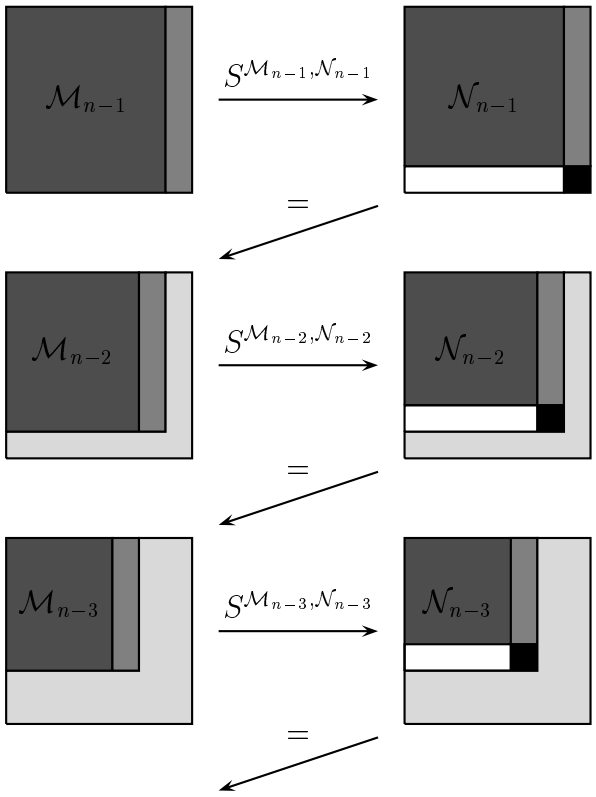

Figure 5: Flow of Algorithm *Determinant*

If $i \notin I^{\mathcal{N}_i}$, then $D_l^{\mathcal{M}_{i-1}} = D_l^{\mathcal{N}_i}, I_l^{\mathcal{M}_{i-1}} = I_l^{\mathcal{N}_i}$ for all i and thus $S^{\mathcal{M}_{i-1}} = S^{\mathcal{N}_i}$. If k is such that $I_k^{\mathcal{N}_i} = i$, then $D_l^{\mathcal{M}_{i-1}} = D_l^{\mathcal{N}_i}, I_l^{\mathcal{M}_{i-1}} = I_l^{\mathcal{N}_i}$ for $l \neq k$, $D_k^{\mathcal{M}_{i-1}} \leq D_k^{\mathcal{N}_i}$ and $I_k^{\mathcal{M}_{i-1}} < I_k^{\mathcal{N}_i}$ and thus $S^{\mathcal{M}_{i-1}} < S^{\mathcal{N}_i}$. So

$$\sum_{i=1}^{n-1} S^{\mathcal{M}_i,\mathcal{N}_i} = S^{\mathcal{M}_{n-1}} - \sum_{i=2}^{n-1} \left(S^{\mathcal{N}_i} - S^{\mathcal{M}_{i-1}}\right) - S^{\mathcal{N}_1} \leq S^{\mathcal{M}_{n-1}}.$$

Since $S^{\mathcal{M}_{n-1}} = O(n^2 d)$, the theorem follows. □

4. The Hermite form

In this section we show how Algorithm *Determinant* can be adjusted to compute the Hermite form (MacDuffee (1956); Newman (1972)) of a full column rank polynomial matrix. By first computing the weak Popov form of such a matrix, we can treat the remaining non–zero rows as a nonsingular matrix.

In Algorithm *Determinant* we ignored the last columns of the matrix when applying simple transformations. If instead we apply all simple transformations

algorithm TriangularForm
input: $\mathcal{T} \in F[x]^{n\times m}$ with full column rank.
output: $\bar{\mathcal{T}}$ in triangular form, left equivalent to $\mathcal{T}$.
$\bar{\mathcal{T}}$:=WeakPopovForm($\mathcal{T}$);
for i **from** $m-1$ **by** -1 **to** 1 **do**
    $\mathcal{M}$:= first i columns of $\bar{\mathcal{T}}$;
    $\mathcal{V}$:= last $m-i$ columns of $\bar{\mathcal{T}}$;
    $\left[\begin{array}{c|c} \mathcal{N} & \mathcal{W} \end{array}\right]$:= ExtendedWeakPopovForm($\mathcal{M}, \mathcal{V}$);
    $\bar{\mathcal{T}} := \left[\begin{array}{cc} \mathcal{N} & \mathcal{W} \end{array}\right]$
od;
Permute rows of $\bar{\mathcal{T}}$ to make $\bar{\mathcal{T}}$ upper triangular;
return $\bar{\mathcal{T}}$

Figure 6: Algorithm *TriangularForm*

on the whole matrix, we end up with a matrix in upper triangular form. This leads to the algorithm in Figure 6. Figure 7 is a pictorial representation of Algorithm *TriangularForm*. One could finally use the black diagonal entries to lower the degree of all off–diagonal entries, yielding a matrix in Hermite Form. The problem with Algorithm *TriangularForm* is that the degrees of the off–diagonal entries in $\mathcal{V}$ may become too high, thus leading to a bad complexity. In order to avoid these high degrees we will apply, during the course of the algorithm, extra elementary transformations that decrease the degrees of these off–diagonal entries. Figure 8 gives a description of Algorithm *HermiteForm* to transform a full column rank matrix into Hermite form. The details of steps (1), (2) and (3) will be explained shortly. Figure 9 is a pictorial representation of Algorithm *HermiteForm*. Here the dark gray columns represent $\mathcal{M}$, the middle gray columns represent $\mathcal{V}$ and the light gray columns represent $\mathcal{A}$.

Figure 10 represents (up to row permutation) the actions of one iteration during the inner while–loop. Here $D_s = D_s^{\mathcal{M}}$ and for $j > i$, d_j is the degree of the bullet entry in column j. The idea is to let $\left[\begin{array}{c|cc} \mathcal{M} & \mathcal{V} & \mathcal{A} \end{array}\right]$ always have the following property.

PROPERTY 1: *For $j > i+1$ and $s < j$ the degree of entry (s,j) of $\left[\begin{array}{c|cc} \mathcal{M} & \mathcal{V} & \mathcal{A} \end{array}\right]$ is at most $D_s + d_j$.*

So Property 1 ensures that the degrees of the entries in the light gray area are not too big. Note that for $s > i+1$ we have $D_s = -1$ and thus when $\left[\begin{array}{c|cc} \mathcal{M} & \mathcal{V} & \mathcal{A} \end{array}\right]$ has Property 1, then for $i+1 < s < j$ the degree of entry (s,j) is less than d_j. This means that the lower triangular part of $\mathcal{A}$ is in Hermite form, and at the end of Algorithm *HermiteForm* $\left[\begin{array}{cc} \mathcal{V} & \mathcal{A} \end{array}\right]$ is in Hermite form.

Suppose $\mathcal{E} = \left[\begin{array}{c|cc} \mathcal{M} & \mathcal{V} & \mathcal{A} \end{array}\right]$ has property 1 and let $\mathcal{F} = \left[\begin{array}{c|cc} \mathcal{N} & \mathcal{W} & \mathcal{B} \end{array}\right]$ be the matrix we get after applying on $\left[\begin{array}{c|cc} \mathcal{M} & \mathcal{V} & \mathcal{A} \end{array}\right]$ the simple transformation of the first kind from row k on row l. Let $\bar{D}_l = D_l^{\mathcal{N}}$. For $j > i+1$ we have by Lemma 3.2 $\deg(\mathcal{F}_{l,j}) \le \bar{D}_l + d_j + D^{\mathcal{M},\mathcal{N}} = D_l + d_j$. So if $\bar{D}_l = D_l$, $\left[\begin{array}{c|cc} \mathcal{N} & \mathcal{W} & \mathcal{B} \end{array}\right]$ still

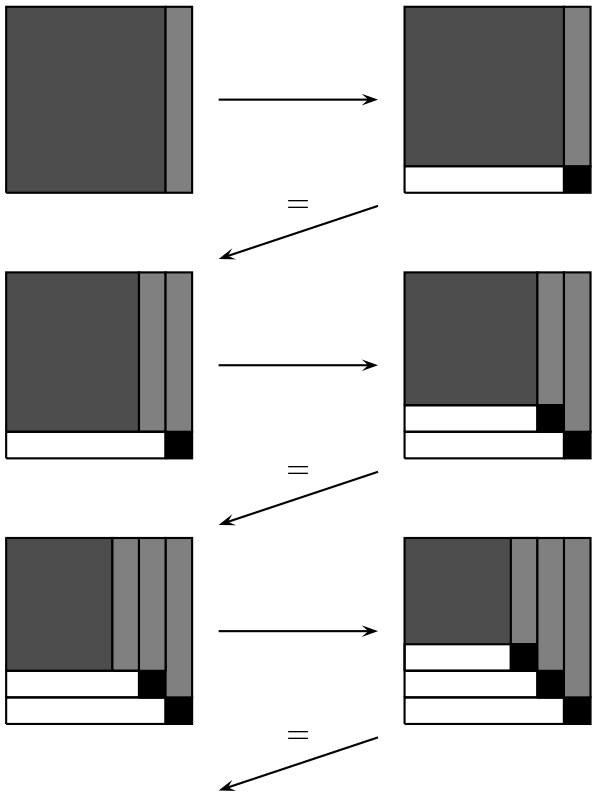

Figure 7: Flow of Algorithm *TriangularForm*

has property 1 and nothing has to be done in step (1). If however $\bar{D}_l < D_l$, the entries in the lth row of $\left[\begin{array}{c|cc} \mathcal{N} & \mathcal{W} & \mathcal{B} \end{array}\right]$ may violate Property 1 and we have to restore the property in step (1). Let q be the quotient of $\mathcal{F}_{l,i+2}$ by $\mathcal{F}_{i+2,i+2}x^{\bar{D}_l+1}$, i.e. $\deg(\mathcal{F}_{l,i+2} - q\mathcal{F}_{i+2,i+2}x^{\bar{D}_l+1}) \leq \bar{D}_l + \deg(\mathcal{F}_{i+2,i+2})$. Then $\deg(q) < D_l - \bar{D}_l$. Let $\mathcal{G}$ be the result of subtracting q times row $i+2$ from row l in $\mathcal{F}$. Then the $(l, i+2)$ entry of $\mathcal{G}$ has degree at most $\bar{D}_l + \deg(\mathcal{G}_{i+2,i+2})$ and thus satisfies Property 1. Moreover, for $j > i+2$

$$\begin{aligned} \deg(\mathcal{G}_{l,j}) &\leq \max(\deg(\mathcal{F}_{l,j}), \deg(q) + \deg(\mathcal{F}_{i+2,j})) \\ &\leq D_l + d_j \end{aligned}$$

since $\deg(\mathcal{F}_{l,j}) \leq D_l + d_j$, $\deg(q) < D_l$ and $\deg(\mathcal{F}_{i+2,j}) \leq d_j - 1$. Subtracting in a similar way in sequence multiples of rows $i+3, \ldots, n$ from row l, we restore Property 1 for row l in step (1).

Now we describe step (2). Figure 11 represents (up to row permutation) the situation just after the while–loop has completed. Before we enlarge $\mathcal{A}$ with column $\mathcal{V}$, we make sure that the entries in $\mathcal{V}$ satisfy Property 1, i.e. make

algorithm HermiteForm
input: $\mathcal{T} \in F[x]^{n\times m}$ with full column rank.
output: $\mathcal{H}$ in Hermite form, left equivalent to $\mathcal{T}$.
$\bar{\mathcal{T}}$:=WeakPopovForm($\mathcal{T}$);
$\mathcal{M}$:= first $n-1$ columns of $\bar{\mathcal{T}}$;
$\mathcal{V}$:= last column of $\bar{\mathcal{T}}$;
$\mathcal{A}$:= empty matrix;
for i **from** $n-1$ **by** -1 **to** 1 **do**
  while $\mathcal{M}$ is not in weak Popov form **do**
    Apply a simple transformation of the first kind on $\left[\begin{array}{c|cc} \mathcal{M} & \mathcal{V} & \mathcal{A} \end{array}\right]$, say from row k
    on row l;
    (1) Use $\bullet$–entries to lower degrees of entries in lth row of $\mathcal{A}$
  od;
  (2) Use $\clubsuit$–entry to lower degrees in $\mathcal{V}$;
  Let l such that $I_l^{\mathcal{M}} = i$;
  $\mathcal{M}$:= first $i-1$ columns of $\left[\begin{array}{ccc} \mathcal{M} & \mathcal{V} & \mathcal{A} \end{array}\right]$;
  $\mathcal{V}$:= ith column of $\left[\begin{array}{ccc} \mathcal{M} & \mathcal{V} & \mathcal{A} \end{array}\right]$;
  $\mathcal{A}$:= last $n-i$ columns of $\left[\begin{array}{ccc} \mathcal{M} & \mathcal{V} & \mathcal{A} \end{array}\right]$;
  (3) Use $\bullet$ entries to lower degrees of entries in lth row of $\mathcal{A}$
od;
$\mathcal{H} := \left[\begin{array}{cc} \mathcal{V} & \mathcal{A} \end{array}\right]$ with rows permuted to make it upper triangular;
Multiply rows of $\mathcal{H}$ with constants to make diagonal entries monic;
return $\mathcal{H}$

Figure 8: Algorithm *HermiteForm*

$\deg(\mathcal{V}_l) \le D_l + d_{i+1}$. Step (2) takes care of this. We could apply row transformations like in step (1), using the $\clubsuit$– and $\bullet$–entries, for this. However, this would be too costly.

Let s be the maximum degree excess in column $\mathcal{V}$, that is, for $1 \le l \le i$ we have $\deg(\mathcal{V}_i) \le D_l + d_{i+1} + s$. Let $\mathcal{Q}^0$ be the $(i+1)$th row of $\mathcal{E}$. For $u = 1, \ldots, s$ let $\mathcal{Q}^u$ be the row vector we get by multiplying $\mathcal{Q}^{u-1}$ by x and, like in step (1), reducing all entries from left to right using rows $i+2, \ldots, n$ of $\mathcal{E}$. Then $\deg(\mathcal{Q}^u_{i+1}) = d_{i+1} + u$ and $\deg(\mathcal{Q}^u_j) < d_j$ for $j > i+1$. Now we can add appropriate monomial multiples of the $\mathcal{Q}^u$ to rows $1, \ldots, i$ to make the entries of $\mathcal{V}$ satisfy Property 1. Notice that this does not destroy Property 1 for the entries in $\mathcal{A}$.

Finally, we describe step (3). When the last column of $\mathcal{M}$ is deleted before we enter the while–loop again, $D_l^{\mathcal{M}}$ may decrease and thus the entries in the lth row may violate Property 1. In step (3) we then apply the same procedure as in step (1) to make sure that the entries in row l satisfy the property again.

THEOREM 4.1: *When d is a bound on the degrees of the entries of $\mathcal{T}$, then the cost of Algorithm* HermiteForm *is bounded by $O(nm^2d^2)$ field operations.*

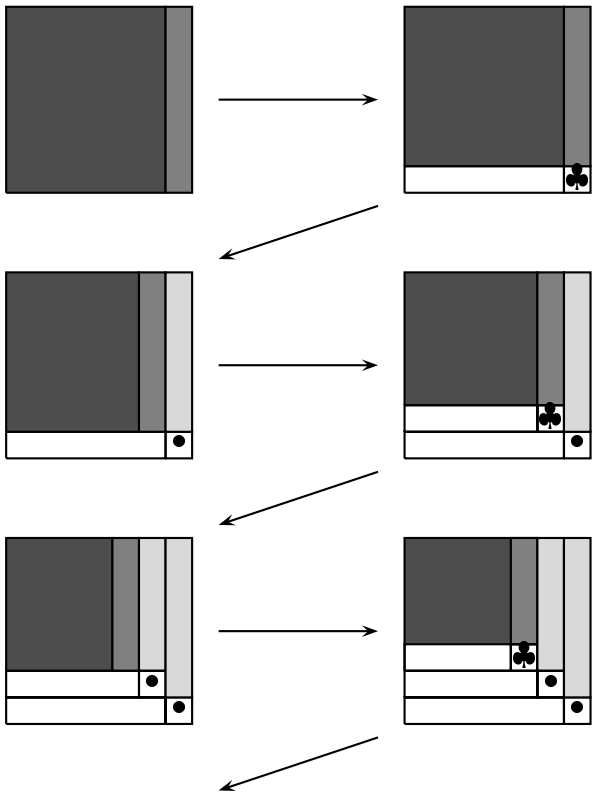

Figure 9: Flow of Algorithm *HermiteForm*

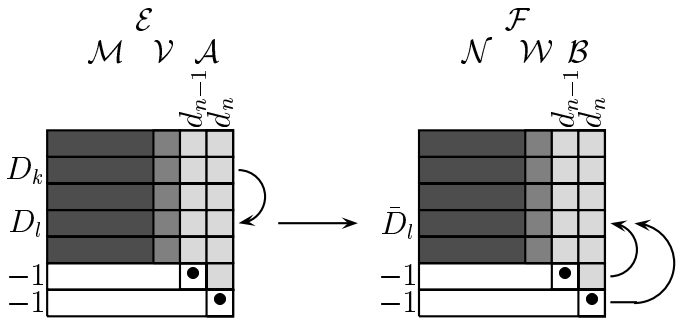

Figure 10: One iteration during **while** –loop

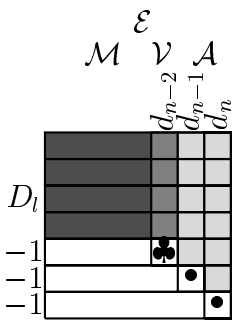

Figure 11: Snapshot after **while** –loop

Proof: By Theorem 2.2 computing $\bar{\mathcal{T}}$ can be accomplished in the allotted time.

By Corollary 2.1 the degrees of the entries in $\mathcal{M}$ are bounded by d. By Lemma 3.2 the entries in $\mathcal{V}$ have degree bounded by $O(md)$. The sum of the degrees of the entries in one row of $\mathcal{A}$ is at most $\sum_{j=i+2}^{m}(d+d_j)$. Since the product of all $\bullet$–entries divides the determinant of $\bar{\mathcal{T}}$ we have $\sum_{j=i+1}^{m}(d+d_j) = O(md)$.

As in the proof of Theorem 3.2 we see that the number of simple transformations applied is bounded by $S^{\bar{\mathcal{T}}} = O(m^2d)$. One simple transformation costs $O(md)$ and thus the cost of all simple transformations is $O(m^3d^2)$.

Adding a multiple of a row like in step (1) costs $O((D_l - \bar{D}_l)md)$ and thus performing step (1) once takes $O((D_l - \bar{D}_l)m^2d)$. Since the total degree drop, i.e. the sum of all $D_l - \bar{D}_l$ is at most md, the total cost of all steps (1) is $O(m^3d^2)$.

The cost of performing step (2) once is bounded by $O(sm^2d)$. By Lemma 3.2 s is bounded by the sum of d and the degree drop during the last invocation of the while–loop. So the sum of all s during the algorithm is $O(md)$ and thus the total cost of all steps (2) is $O(m^3d^2)$.

As in step (1), the cost of step (3) is bounded by $O(m^3d^2)$ and making the diagonal entries monic can be accomplished in $O(m^2d)$. □

Notice that Algorithm *HermiteForm* can easily be adjusted to compute the Hermite form of $\mathcal{M}$ when $\mathcal{M}$ has not full column rank. However, the degrees of some off–diagonal entries can become very large in that case, thus leading to a worse complexity.

Other approaches to Hermite form computation

Many algorithms have been proposed to compute the Hermite form. They can be classified as follows:

1. Direct methods, applying elementary row transformations over the ring. To ensure a polynomial running time the potential for exponential growth of intermediate entries (see Fang and Havas (1997)) has to be avoided (Chou and Collins (1982); Havas et al. (1998); Kannan and Bachem (1979)). See also Havas and Majewski (1994).

2. Modulo determinant methods, performing arithmetic modulo a multiple of a determinant. The modular arithmetic avoids exponential growth of intermediate expressions (Domich (1989); Domich et al. (1987); Hafner and McCurley (1991); Iliopoulos (1989); Storjohann and Labahn (1996)).
3. Coefficient methods for polynomial matrices, translating the Hermite problem to a problem over the coefficient ring (Bitmead et al. (1978); Labhalla et al. (1996); Villard (1996)).

Some of these algorithms use fast matrix and/or fast polynomial arithmetic to improve their complexity. The best known complexity result is $O(n^\theta(nd)^{1+\epsilon})$ field operations, where θ is the exponent for matrix multiplication (Storjohann and Labahn (1996)).

The algorithm we present here belongs to the first class. Unlike most methods in this class we do not follow the greedy approach of Gaussian elimination to triangularize the input matrix. Instead we have used lattice reduction *a la* the Popov form to ensure good bounds on the degrees of intermediate polynomials. Basis reduction is applied to the problem of computing the Hermite form of an integer matrix in Havas et al. (1998). Most direct methods triangularize the input matrix from the left top to the right bottom. The algorithm we describe here takes the novel approach of recovering H row by row starting with the last row and proceeding up to the first. Polynomial degree being non–Archimedean is one of the reasons for the good degree bounds we get.

More recently, a general framework for normal forms of polynomial matrices is given in Beckermann et al. (1999), where it is shown that Hermite form can be recovered from the Popov form of a well chosen "shift" of the input matrix. There a major concern is the complexity of coefficients which is handled by fraction–free methods.

5. Triangular factorization

In this section, we show how the previous algorithms can be used to obtain a triangular factorization of a full column rank matrix $T \in F[x]^{n\times m}$, that is, find a matrix H in Hermite form and a unimodular matrix U such that $T = UH$.

The first step is to determine m independent rows of T. For this let M_1 be the first row of T and let M_{i+1} be the weak Popov form of the matrix M_i with row $i+1$ of T added. The rows that make the number of non–zero rows in M_i increase then constitute a set of m independent rows of T.

For simplicity assume that the first m rows of T are independent. Append to T an $(n-m)\times(n-m)$ identity matrix to the right bottom, yielding $\bar{T}$, i.e.

$$\bar{T} = \left[\begin{array}{c|c} T & \begin{array}{c} 0 \\ I \end{array} \end{array}\right].$$

Now compute the Hermite form $\bar{H}$ of $\bar{T}$ with Algorithm *HermiteForm* and

$U = \bar{T}\bar{H}^{-1}$. If

$$\bar{H} = \left[\begin{array}{c|c} H & * \\ 0 & * \end{array}\right],$$

then

$$T = U \left[\begin{array}{c} H \\ 0 \end{array}\right]$$

is a triangular factorization of T.

THEOREM 5.1: *When $T \in F[x]^{n\times m}$ has rank m and entries of degree bounded by d, one can compute the triangular factorization of T in $O(n^3d^2)$ field operations. Moreover, the degrees of the entries of the unimodular transformation matrix are bounded by d.*

Proof: Use the algorithm described before. Determining the m independent rows has the cost of computing a weak Popov form of T which costs $O(nm^2d^2)$ by Theorem 2.2. Computing the Hermite form of $\bar{T}$ costs $O(n^3d^2)$ by Theorem 4.1. Finally, compute the rows of U one by one by backsolving. Since $\bar{H}$ is in Hermite form the sum of the degrees of all entries in one row is bounded by $\deg(\det(\bar{H})) = O(nd)$. Moreover, it follows that the degree of the entries in U are bounded by d. Thus, computing one row of U costs $O(n^2d^2)$ and computing U can be accomplished in the allotted time. □

6. Polynomial linear system solving

In this section, we show how the previous algorithms can be used to solve a polynomial linear system of equations in the following sense:

1. If the system has no rational solution, that is no solution over $F(x)$, state this;
2. If the system has a Diophantine solution, that is a solution over $F[x]$, give one;
3. If the system has a rational solution but no Diophantine solution, give a rational solution.

Let $M \in F[x]^{n\times m}$ and $b \in F[x]^{1\times m}$. We want to solve the system $xM = b$.

First we compute the rank r of M and the rank of M with b attached. If these ranks differ, then the system has no rational solution.

Otherwise we determine r independent columns of M using the technique from Section 5. Let N consist of those r independent columns and c consist of the corresponding entries in b. We then only have to consider the system $xN = c$.

Next we compute the triangular factorization

$$N = U \left[\begin{array}{c} H \\ 0 \end{array}\right]$$

of N, where H is nonsingular and in Hermite form. Let $D = \det(H)$, i.e. the product of all diagonal entries of H. Then solve $yH = Dc$ with backsolving and solve $xU = \begin{bmatrix} y & 0 \end{bmatrix}$ with Hensel lifting. Then x/D is the desired result.

THEOREM 6.1: *When $n \geq m$ and d is a bound on the degrees of the entries of M and b, then one can solve the linear system $xM = b$ in $O(n^3d^2)$ field operations.*

Proof: Use the algorithm described above. Computing r costs $O(nmrd^2)$ by Corollary 2.3. Determining r independent columns of A costs $O(nmrd^2)$ by Theorem 2.2. Computing the triangular factorization of N costs $O(n^3d^2)$ by Theorem 5.1. Since $\deg(D) \leq rd$, computing D and Dc can be accomplished in $O(r^2d^2)$ field operations. By Cramers rule (Horn and Johnson (1985)), the degree of all entries in y is $O(rd)$. Since H is in Hermite form, the sum of degrees of all entries in one row of H is bounded by rd. It follows that solving $yH = Db$ by backsolving costs $O(r^3d^2)$. By Theorem 5.1 the degrees of the entries of U are bounded by d. By Mulders and Storjohann (1999, Theorem 22) it then follows that solving $xU = \begin{bmatrix} y & 0 \end{bmatrix}$ can be accomplished in $O(n^3d^2)$ field operations. Finally, computing x/D costs $O(rn^2d^2)$. □

The algorithm described in this section is deterministic. If we allow probabilistic methods, solving a linear system in the sense above can be accomplished in $O(nmrd^2)$ field operations (Mulders and Storjohann (2000a)).

7. The Popov form

In this section, we show how we can transform a matrix that is in weak Popov form into Popov form. Combined with Algorithm *WeakPopovForm* this will yield an algorithm to transform any matrix into Popov form.

In Kailath (1980) and Villard (1996) the Popov form of a matrix is computed via translation to problems over F with bigger dimensions. Here we will improve the complexity obtained in Villard (1996) for this problem. In Beckermann et al. (1999) fraction free methods are introduced when computing the Popov form. The main concern there is to avoid intermediate coefficient swell in the case of say integer polynomial matrices. Here we only consider arithmetic complexity, that is, only count field operations.

Definition: M is said to be in ascending order if for $i < l$ we have $D_i^M < D_l^M$ or $(D_i^M = D_l^M \neq -1$ and $I_i^M < I_l^M)$.

Note that when M is in ascending order, the zero rows of M are on top, i.e. have smallest row index.

Definition (see also Kailath (1980)): M is said to be in Popov form if

1. M is in weak Popov form;
2. M is in ascending order;

3. P_i^M is monic for $i \in C^M$;
4. $\deg(m_{i,I_l^M}) < D_l^M$ for $l \in C^M$ and $i \neq l$.

When M is in weak Popov form we can transform M into ascending order by permuting the rows of M.

Assume that M already satisfies properties 1 and 2. We will make M satisfy property 4 by applying simple transformations of the second kind on M. In order that a simple transformation does not cancel progress made before, we apply the simple transformations in a particular order.

Suppose that the first $k-1$ rows of M already satisfy property 4, that is $\deg(m_{i,I_l^M}) < D_l^M$ for $l \in C^M$, $i \neq l$ and $i, l < k$.

If the kth row of M is the zero row, then the first k rows of M are all zero rows and satisfy property 4.

Now suppose that the kth row of M is not the zero row. For $i < k$ we then have:

1. If $D_i^M = -1$, then $\deg(m_{i,I_k^M}) = -\infty < D_k^M$.
2. If $D_i^M < D_k^M$, then $\deg(m_{i,I_k^M}) \leq D_i^M < D_k^M$.
3. If $D_i^M = D_k^M$, then $I_i^M < I_k^M$ and thus $\deg(m_{i,I_k^M}) < D_i^M = D_k^M$.

So $\deg(m_{i,I_k^M}) < D_k^M$ for $i < k$ and we only have to make the entries in row k satisfy property 4.

Let $\delta^M = \max_{i<k, i\in C^M}(\deg(m_{k,I_i^M}) - D_i^M)$. If $\delta^M < 0$, then the first k rows of M satisfy property 4. Otherwise let $l < k, l \in C^M$ such that $\delta^M = \deg(m_{k,I_l^M}) - D_l^M$ and $N = (n_{i,j})$ the matrix we get when we apply the simple transformation (of the second kind) of row l on row k. By Lemma 2.2 D_k^M and I_k^M do not change and thus N still satisfies properties 1 and 2 and still $\deg(n_{i,I_k^N}) < D_k^N$ for $i < k$.

Let $\delta^N = \max_{i<k, i\in C^N}(\deg(n_{k,I_i^N}) - D_i^N)$. If $\delta^N < 0$, the first k rows of N satisfy property 4. Otherwise, let $\nu^M = \#\{i < k \mid \delta^M = \deg(m_{k,I_i^M}) - D_i^M\}$ and $\nu^N = \#\{i < k \mid \delta^N = \deg(n_{k,I_i^N}) - D_i^N\}$. We now show that $(\delta^N, \nu^N) < (\delta^M, \nu^M)$ in the lexicographic order. For this we only have to show that $\delta^N \leq \delta^M$ and if $\delta^N = \delta^M$, then $\nu^N < \nu^M$.

For $i < k$ such that $i \neq l$ and $i \in C^N$ let $j = I_i^N = I_i^M$ and note that $D_i^N = D_i^M$. Then

$$\deg(n_{k,j}) - D_i^N \leq \max(\deg(m_{k,j}) - D_i^N, \delta^M + \deg(m_{l,j}) - D_i^N).$$

Since the first $k-1$ rows of M already satisfy property 4 we have $\deg(m_{l,j}) - D_i^N < 0$. So if $\deg(m_{k,j}) - D_i^M < \delta^M$, then $\deg(n_{k,j}) - D_i^N < \delta^M$; if $\deg(m_{k,j}) - D_i^M = \delta^M$, then $\deg(n_{k,j}) - D_i^N = \delta^M$. Moreover, $\deg(n_{k,I_l^N}) - D_l^N < \deg(m_{k,I_l^N}) - D_l^N = \delta^M$, since we applied the simple transformation of row l on row k. We see that either ($\delta^N = \delta^M$ and $\nu^N = \nu^M - 1$) or $\delta^N < \delta^M$.

Figure 12 describes an algorithm to compute the Popov form of a matrix based on our previous observations.

algorithm PopovForm
input: $\mathcal{M} \in F[x]^{n\times m}$.
output: $\mathcal{N}$ in Popov form, left equivalent to $\mathcal{M}$.
$\mathcal{A}$:= WeakPopovForm($\mathcal{M}$);
Permute rows of $\mathcal{A}$ such that $\mathcal{A}$ is in ascending order;
for k to n **do**
  if kth row is not the zero row **then**
    do
      Let $\delta = \max_{i<k, i\in C^{\mathcal{A}}}(\deg(m_{k,I_i^{\mathcal{A}}}) - D_i^{\mathcal{A}})$;
      if $\delta < 0$ **then**
        break
      fi;
      Let $l < k, l \in C^{\mathcal{A}}$ such that $\deg(m_{k,I_l^{\mathcal{A}}}) - D_l^{\mathcal{A}} = \delta$;
      Apply simple transformation of row l on row k
    od
  fi
od;
Multiply non–zero rows of $\mathcal{A}$ with constant to make pivots monic;
$\mathcal{N}$:=copy($\mathcal{A}$);
return $\mathcal{N}$

Figure 12: Algorithm *PopovForm*

THEOREM 7.1: *Algorithm* PopovForm *is correct. If d is a bound on the degrees of the entries in $\mathcal{M}$ and r is the rank of $\mathcal{M}$, then the cost of Algorithm* PopovFrom *is bounded by $O(nmrd^2)$ field operations.*

Proof: Since always $\delta^M \le d$ and $\nu^M < r$ it follows from the previous observations that in the loop at most $O(rd)$ simple transformations are applied on each non–zero row. So the total number of simple transformations applied in the loop is $O(r^2d)$. From Lemma 2.2 it follows that the degrees of all entries in $\mathcal{A}$ are always bounded by d. Thus the cost of the loop is $O(r^2md^2)$. The theorem now follows from Theorem 2.2. □

8. Reduced bases

In von zur Gathen (1984) the notion of reduced basis is introduced. For a polynomial matrix $M = (m_{i,j}) \in F[x]^{n\times m}$ of rank r this boils down to the following.

Definition: M is said to be *reduced* if

1. Rows $r+1, \ldots, n$ are zero rows;
2. For $1 \le i \le r$ we have $\deg(m_{i,k}) < \deg(m_{i,i})$ for $1 \le k < i$ and $\deg(m_{i,k}) \le \deg(m_{i,i})$ for $i \le k \le m$;
3. $\deg(m_{i,i}) \le \deg(m_{j,j})$ for $1 \le i \le j \le r$.

In von zur Gathen (1984) and von zur Gathen and Gerhard (1999, Exercise 16.12) an algorithm is described to transform a full row rank matrix, up to column permutation, into a reduced matrix by a unimodular row transformation. The complexity of this algorithm turns out to be $O(mn^3d^{2+\epsilon})$ field operations.

Now suppose M is already in Popov form. If $\deg(P_k^M) \leq \deg(P_l^M)$ for $k \neq l$, then $\deg(m_{l,I_k^M}) < \deg(P_k^M) \leq \deg(P_l^M)$. From this we see that by permuting the rows and columns of M such that the pivots of M end up on the diagonal with increasing degree from top to bottom, we get a reduced matrix. So we can transform any matrix in reduced form by first computing its Popov form and then permuting its rows and columns. The cost of this is $O(nmrd^2)$ by Theorem 7.1, which is one order of magnitude better than the algorithm described in von zur Gathen (1984).

Reduced bases are used in von zur Gathen (1984) to compute short vectors in modules. In the polynomial case the weak Popov form already suffices for that.

LEMMA 8.1: *If M is in weak Popov form and l is such that* $\deg(P_l^M) = \min_{1\leq i\leq n}(\deg(P_i^M))$, *then all vectors in the $F[x]$–module generated by the rows of M have degree at least* $\deg(P_l^M)$.

Proof: Let $r^i \in F[x]^{1\times m}$ denote the ith row of $M = (m_{i,j})$ and let $d_i \in F[x]$ such that $r = \sum_{i=1}^n d_i r^i \neq 0$. Let k such that $\deg(d_k P_k^M)$ is maximal and I_k^M maximal, i.e. for $i \neq k$ either $\deg(d_i P_i^M) < \deg(d_k P_k^M)$ or $(\deg(d_i P_i^M) = \deg(d_k P_k^M)$ and $I_i^M < I_k^M$. Then for $i \neq k$ we have

1. if $\deg(d_i P_i^M) < \deg(d_k P_k^M)$, then $\deg(d_i m_{i,I_k^M}) \leq \deg(d_i P_i^M) < \deg(d_k P_k^M)$;
2. if $\deg(d_i P_i^M) = \deg(d_k P_k^M)$ and $I_i^M < I_k^M$, then $\deg(d_i m_{i,I_k^M}) < \deg(d_i P_i^M) = \deg(d_k P_k^M)$.

It follows that $\deg(r_{I_k^M}) = \deg(d_k P_k^M) \geq \deg(P_l^M)$ □

9. Discrete valuation rings

In this section we extend the notion of weak Popov form to the setting of discrete valuation rings.

Definition: (Atiyah and MacDonald (1969)) Let K be a field. A *discrete valuation* on K is a mapping v of K^* onto $\mathbb{Z}$ such that

1. $v(ab) = v(a) + v(b)$;
2. $v(a+b) \geq \min(v(a), v(b))$.

Let R be the ring consisting of 0 and all $a \in K^*$ such that $v(a) \geq 0$. Then R is called a *discrete valuation ring*. R is a local ring and its maximal ideal $\mathcal{I}$ is the set of all $a \in K$ such that $v(a) > 0$. Let $u \in R$ such that $v(u) = 1$. Then $\mathcal{I} = (u)$, the ideal of R generated by u. The set R^* of units of R is the set of all $a \in K$ such that $v(a) = 0$. Let $S \subseteq R^* \cup \{0\}$ such that the canonical projection map $S \to R/\mathcal{I}$

is a bijection. For $a, b \in R$ with $v(a) \geq v(b)$, we have $v(u^{v(b)-v(a)}a/b) = 0$, so there exists a unique $c \in S \setminus \{0\}$ such that $u^{v(b)-v(a)}a/b - c \in \mathcal{I}$, and thus

$$v(a - cu^{v(a)-v(b)}b) > v(a). \tag{4}$$

EXAMPLE 2: *Let F be a field. The set $F[[x]]$ of formal power series in x is a discrete valuation ring. For $a \in F[[x]]$, $v(a)$ is the maximum $n \in \mathbb{N}$ such that x^n divides a. For S we can take F in this case.*

Let $M = (m_{i,j}) \in R^{n \times m}$. As an analogue to Section 2 we define the pivot element P_i^M of row i of M as the rightmost element with minimum valuation in its row, the pivot index I_i^M as the index of P_i^M, i.e. $P_i^M = m_{i,I_i^M}$, and the pivot valuation D_i^M as $v(P_i^M)$. Again, M is said to be in weak Popov form if all (non–zero) indices are different. If $v(m_{l,I_k^M}) \geq v(P_k^M)$, let $c \in S \setminus \{0\}$ such that $v(m_{l,I_k^M} - cu^{v(m_{l,I_k^M})-v(P_k^M)}P_k^M) > v(m_{l,I_k^M})$. Then we call subtracting $cu^{v(m_{l,I_k^M})-v(P_k^M)}$ times row k from row l the simple transformation of row k on row l. The analogue of Lemma 2.2 holds also.

LEMMA 9.1: *Let N be the matrix we get after applying the simple transformation of row k on row l of M. Then $I_i^N = I_i^M, D_i^N = D_i^M$ for $i \neq l$ and $D_l^N \geq D_l^M$. If the transformation is of the first kind, then either $D_l^N > D_l^M$ or ($D_l^N = D_l^M$ and $I_l^N < I_l^M$). If the transformation is of the second kind, then $I_l^N = I_l^M$ and $D_l^N = D_l^M$.*

Now we can apply Algorithm *WeakPopovForm* to transform M into weak Popov form. However, the algorithm may run forever as the following example shows.

EXAMPLE 3: *For*

$$\mathcal{M} = \begin{bmatrix} \frac{x}{1-x} \\ 1 \end{bmatrix} = \begin{bmatrix} x + x^2 + x^3 + \cdots \\ 1 \end{bmatrix} \in F[[x]]^{2 \times 1}$$

Algorithm WeakPopovForm *will keep on subtracting x^i from $\mathcal{M}_{1,1}$ for increasing i and thus run forever. However, it is possible to transform $\mathcal{M}$ into weak popov form by a unimodular transformation, since*

$$\begin{bmatrix} 1 & 1 \\ -x & 1-x \end{bmatrix} \begin{bmatrix} 1 \\ x + x^2 + x^3 + \cdots \end{bmatrix} = \begin{bmatrix} 1 + x + x^2 + \cdots \\ 0 \end{bmatrix}.$$

Notice that the unimodular transformation matrix is even over $F[x]$. Besides, Algorithm WeakPopovForm *only computes transformations over $F[x]$.*

Lemma 2.3 and Corollary 2.2 are still valid in the discrete valuations ring setting and thus the number of different values that a pivot index can assume during the course of Algorithm *WeakPopovForm* is bounded by the rank of the matrix. The following lemma shows that the algorithm still works when $\mathcal{M}$ has full row rank.

THEOREM 9.1: *Suppose $\mathcal{M}$ has full row rank. Let d be the valuation of the determinant of some nonsingular $n \times n$ submatrix of $\mathcal{M}$. Then Algorithm* WeakPopovForm *is correct and applies at most $dn + n(n-1)$ simple transformations of the first kind.*

Proof: Since the index of a row can assume at most n different values, Lemma 9.1 implies that the valuation of row l, that is $\min_{1 \le j \le m}(v(\mathcal{M}_{l,j}))$, must have increased after applying n simple transformations of the first kind on row l and so when s_l simple transformations of the first kind are applied on row l the valuation of that row must have increased by at least $\lfloor s_l/n \rfloor$.

Let $\mathcal{G}$ be a nonsingular submatrix of $\mathcal{M}$ and $d = v(\det(\mathcal{G}))$. Suppose that Algorithm *WeakPopovForm* applies more than $dn+n(n-1)$ simple transformations of the first kind and suppose $\mathcal{G}$ is transformed into $\mathcal{H}$ after applying the first $dn + n(n-1) + 1$ simple transformations. Then $v(\det(\mathcal{H})) \ge \sum_{i=1}^{n} \lfloor s_i/n \rfloor > d$, contradicting $\det(\mathcal{H}) = \det(\mathcal{G})$.

So Algorithm *WeakPopovForm* does stop and is thus correct by Lemma 2.1. □

As in the polynomial case, the weak Popov form in the current setting can be used to determine a vector with minimal valuation in the R–module generated by the rows of a matrix.

The analogue of Popov form would insists that $v(m_{i,I_l^M}) > D_l^M$ for $i \ne l$. It is in general not possible to transform a matrix into Popov form by only using unimodular transformations.

EXAMPLE 4: *Let*

$$M = \begin{bmatrix} 1 & x \\ x^2 & x^2 \end{bmatrix}.$$

Then M is nonsingular and in weak Popov form. Suppose

$$U = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \in F[[x]]^{2\times 2}$$

is unimodular and $N = UM$ is in weak Popov form. We may assume (eventually switch rows) that $v(a) = 0$. Then $v(N_{1,1}) = 0$, $v(N_{1,2}) = 1$ and $v(N_{2,2}) \ge 1$. So $I_1^N = 1$ and thus I_2^N must be 2. Since $v(N_{1,2}) \le v(N_{2,2})$, N cannot be in Popov form.

10. Conclusions

We have introduced the notion of weak Popov form for a polynomial matrix and described a simple algorithm for transformation to the form. The algorithm applies only simple transformations which avoid growth of intermediate expressions; this leads to a complexity of $O(nmrd^2)$ field operations for an input matrix $M \in F[x]^{n\times m}$ of rank r whose entries have degrees bounded by d.

The algorithm is central to various other algorithms: i.e. for determinant, Hermite form, Popov form, Diophantine solution and short vector computation.

We also extended the notion of weak Popov form to the setting of discrete valuation rings. Such an extension does not seem possible for the notion of Popov form.

The analysis in this paper only counts field operations and thus gives a good estimate of the cost of the algorithms when F is a finite field. The constants involved in the complexity results are not explicitly computed but can easily be derived. Since these constants are small the algorithms will perform well in practice. Some comparative experiments with implementations of various algorithms in Aldor (Watt et al. (1994)) confirm this.

The performance of the algorithms for other coefficient fields F, e.g. $F = \mathbb{Q}$ (or $\mathbb{Z}$), is another issue. In this case, intermediate expression swell on coefficient level is introduced, leading to a severe breakdown of the algorithms performance. Combining the algorithms with homomorphic imaging schemes may be the solution to this problem. Another idea may be to introduce fraction free techniques, similar as in Beckermann et al. (1999), in the algorithms of this paper. Further research needs to be done in this area.

Another remaining question is how to transform in the discrete valuation ring setting a non full row rank matrix into weak Popov form. The algorithm presented in Section 9 may run forever on such a matrix.

References

M.F. Atiyah and I.G. MacDonald. *Introduction to Commutative Algebra.* Addison–Wesley, 1969.

Bernhard Beckermann, George Labahn, and Gilles Villard. Shifted normal forms of polynomial matrices. In Sam Dooley, editor, *ISSAC99*, pages 189–196, 1999.

R.R. Bitmead, S.Y. Kung, B.D.O. Anderson, and T. Kailath. Greatest common divisors via generalized Sylvester and Bezout matrices. *IEEE Trans. Automat. Control.*, 23(6):1043–1047, 1978.

Tsu-Wu J. Chou and George E. Collins. Algorithms for the solutions of systems of linear Diophantine equations. *SIAM Journal of Computing*, 11:687–708, 1982.

P. D. Domich. Residual Hermite normal form computations. *ACM Trans. Math. Software*, 15:275–286, 1989.

P. D. Domich, R. Kannan, and L. E. Trotter, Jr. Hermite normal form computation using modulo determinant arithmetic. *Mathematics of Operations Research*, 12(1):50–59, 1987.

Xin Gui Fang and George Havas. On the worst-case complexity of integer gaussian elimination. In Wolfgang W. Küchlin, editor, *ISSAC97*, pages 28—31, 1997.

James L. Hafner and Kevin S. McCurley. Asymptotically fast triangularization of matrices over rings. *SIAM Journal of Computing*, 20(6):1068–1083, December 1991.

George Havas and Bohdan S. Majewski. Hermite normal form computation for integer matrices. *Congressus Numerantium*, 105:87–96, 1994.

George Havas, Bohdan S. Majewski, and Keith R. Matthews. Extended GCD and Hermite normal form algorithms via lattice basis reduction. *Experimental Mathematics*, 7:125—136, 1998.

Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

Costas S. Iliopoulos. Worst-case complexity bounds on algorithms for computing the canonical structure of finite abelian groups and the Hermite and Smith normal forms of an integer matrix. *SIAM Journal of Computing*, 18(4):658–669, 1989.

Thomas Kailath. *Linear Systems*. Prentice Hall, 1980.

Ravindran Kannan and Achim Bachem. Polynomial algorithms for computing the Smith and Hermite normal forms of and integer matrix. *SIAM Journal of Computing*, 8(4):499–507, November 1979.

Salah Labhalla, Henri Lombardi, and Roger Marlin. Algorithmes de calcul de la réduction de Hermite d'une matrice à coefficients polynomiaux. *TCS*, 161: 69—92, 1996.

C. C. MacDuffee. *The Theory of Matrices*. Chelsea, New York, 1956.

Thom Mulders and Arne Storjohann. Diophantine linear system solving. In Sam Dooley, editor, *Proc. ISSAC'99*, pages 181—188, 1999.

Thom Mulders and Arne Storjohann. Certified dense linear system solving. Techreport 355, ETH Zurich, Department of Computer Science, 2000a.

Thom Mulders and Arne Storjohann. Triangular factorization of polynomial matrices. Poster at ISSAC2000, 2000b.

M. Newman. *Integral Matrices*. Academic Press, 1972.

Arne Storjohann and George Labahn. Asymptotically fast computation of Hermite normal forms of integer matrices. In Y. N. Lakshman, editor, *Proc. ISSAC'96*, pages 259–266. ACM Press, 1996.

Gilles Villard. Computing Popov and Hermite forms of polynomial matrices. In Y. N. Lakshman, editor, *Proc. ISSAC'96*, pages 250–258. ACM Press, 1996.

Joachim von zur Gathen. Hensel and newton methods in valuation rings. *Mathematics of Computation*, 42:637—661, 1984.

Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, 1999.

S.M. Watt et al. A First Report on the A$\sharp$ Compiler. In *Proc. ISSAC'94*, pages 25–31, 1994.