The Modulo N Extended GCD Problem for Polynomials*

Thom Mulders Arne Storjohann

Institute of Scientific Computing
ETH Zurich, Switzerland
{mulders, storjoha}@inf.ethz.ch
http://www.inf.ethz.ch/personal/{mulders, storjoha}

Abstract

We study the following problem: Given $a, b, N \in F[x]$ with $\gcd(a, b, N) = 1$ and N nonzero, compute a minimal degree $f \in F[x]$ which satisfies $\gcd(a + fb, N) = 1$. We give a deterministic algorithm for solving this problem that is applicable over any field. The algorithm is designed to solve efficiently a succession of such problems for a fixed N. When $q = \#F > \deg N$ the solution will satisfy $\deg f = 0$. When $q \leq \deg N$ we conjecture that the solution satisfies $\deg f \leq \lceil \log_q \deg N \rceil$; in this case the complexity bound we give for the algorithm depends on this conjecture.

As an application we demonstrate a deterministic algorithm for computing transforming matrices for the Smith normal form of a nonsingular $A \in F[x]^{n\times n}$. When q is too small most previous algorithms require working over an algebraic extension of F and may not produce transforming matrices over $F[x]$. The algorithm we propose will produce transforming matrices over $F[x]$, for fields F of any size.

1 Introduction

Let F be a field and N a nonzero polynomial from $F[x]$. The restricted modulo N extended gcd problem for polynomials takes as input $a, b \in F[x]$ with $\gcd(a, b, N) = 1$, and asks for a minimal degree $f \in F[x]$ which satisfies

$$\gcd(a + fb, N) = 1. \qquad (1)$$

When $\#F > \deg N$ it is easy to show that there exists a solution to (1) with $\deg f = 0$, that is, with $f \in F$. When $\#F \leq \deg N$ there may not exist a solution $f \in F$. A standard approach taken when the ground field F is too small is to work over an algebraic extension field K of F which has a sufficient number of elements. For example, the randomized algorithms proposed in [6, 7, 10] for computing normal forms of matrices over $F[x]$ as well as the deterministic normal form algorithm in [12] all take this approach. A drawback of working over K is that the final result may not be over $F[x]$ but over $K[x]$. This may happen, for example, when the algorithms in [7, 12] are used to compute transforming matrices for the Smith normal form over $F[x]$.

In our work we observe that we can avoid algebraic numbers by allowing polynomial solutions to (1) with $\deg f > 0$. In this way we avoid the presence of algebraic numbers in the final result of the computation. The possibility of using polynomials instead of algebraic numbers has also been mentioned in [13, Remark 6.4]. To ensure good degree bounds of the polynomials in the output, we want the solution f to be small. In particular, the restricted modulo N extended gcd problem we have posed asks for a minimal degree solution to (1).

The main contribution of this paper is a deterministic algorithm for solving the restricted modulo N extended gcd problem that is applicable over any field. The algorithm is designed to solve efficiently a succession of such problems for a fixed N. An important feature of the algorithm is output sensitivity; the running time depends on the minimal degree of a solution to (1). To give complexity results we use the function $P(t)$, which can be taken to be $O(M(t) \log t)$, where $M(t)$ bounds the number of field operations required to compute the product of two degree t polynomials from $F[x]$. We distinguish between the following two cases.

Large field case

When $q = \#F > \deg N$ the minimal degree solution to (1) satisfies $\deg f = 0$ and our algorithm requires $O(sP(\deg N) + P(\deg N) \deg N)$ field operations to solve s instances of (1) for a fixed N. This is a satisfying result since for $s \geq \deg N$ the cost is the same as that for computing s pairwise gcds. Note that we assume that each pair of input polynomials have degree bounded by $\deg N$.

Small field case

When $q = \#F \leq \deg N$ we conjecture that the minimal degree solution to (1) satisfies $\deg f \leq \lceil \log_q \deg N \rceil$. If this conjecture is true, then our algorithm has a worst case complexity of $O\tilde{~}(s(\deg N)^2 q + P(\deg N) \deg N)$ field operations for solving s instances of (1) for a fixed N.

We apply our gcd algorithm to the problem of computing transforming matrices for the Smith normal form of a polynomial matrix. Recall the definition of the Smith form. Corresponding to any nonsingular $A \in F[x]^{n\times n}$ there exist unimodular (square and invertible) U and V over $F[x]$ such that

$$UAV = S = \mathrm{diag}(s_1, s_2, \ldots, s_n),$$

[*] This work has been supported by grants from the Swiss Federal Office for Education and Science in conjunction with partial support by ESPRIT LTR Project no. 20244 — ALCOM–IT.

with each s_i nonzero, monic and with $s_i | s_{i+1}$ for $1 \leq i \leq n-1$. Although S is unique, the transforming matrices are highly nonunique and may have large degree entries. The goal is to produce U and V with good degree bounds.

Let $A \in F[x]^{n \times n}$ have degrees of entries bounded by $d = \deg A$. The algorithm in [12] computes U and V satisfying $\deg U, \deg V = O(nd)$. Moreover, the sum of the degrees of entries in V will be bounded by $O(n^2 d)$, meaning that V requires about the same space to write down as the input matrix. These degree bounds are very good. In order to ensure that transforming matrices are over $F[x]$, the algorithm in [12] requires that $\#F > 2nd$. When $d = O(n)$ the cost of the algorithm in [12] is $O\tilde{\ }(n^9)$ field operations assuming pseudo-linear polynomial multiplication.

In this paper we propose an algorithm that returns transforming matrices over $F[x]$ for any field F. Our approach is to adapt the algorithm proposed in [9] to compute small size transforming matrices for the Smith form of an integer matrix. That algorithm was based on computing small solutions to the modulo N extended gcd problem for integers[1], which is replaced by our algorithm for solving instances of the modulo N extended gcd problem over $F[x]$.

When $\#F > nd$ the Smith form algorithm requires $O(n^3 P(nd) + ndP(nd))$ field operations to produce transforming matrices with the same good degree bounds as in [12]. If $d = O(n)$ this complexity simplifies to $O\tilde{\ }(n^5)$ field operations assuming pseudo-linear polynomial multiplication.

When $\#F \leq nd$ and the conjecture is true, then the Smith form algorithm requires $O\tilde{\ }(n^3 P(nd) + ndP(nd) + n^4 d^2 \#F)$ field operations. If $d = O(n)$ this simplifies in the worst case, i.e. $\#F = nd$, to $O\tilde{\ }(n^8)$. For a constant small field F the complexity reduces to $O\tilde{\ }(n^6)$, assuming pseudo-linear polynomial multiplication.

The rest of this paper is organized as follows. In section 2 we present a deterministic algorithm for solving the restricted modulo N extended gcd problem. In section 3 we investigate the function $g_F(N)$ which bounds the degree of a minimal degree solution to (1) and conjecture a bound for $g_F(N)$ in terms of $\deg N$ and $\#F$. In section 4 we give an algorithm for solving a generalization of the modulo N extended gcd problem. In section 5 we use the results presented so far to compute transforming matrices for the Smith form of a nonsingular $A \in F[x]^{n \times n}$.

Throughout we denote by $\mathcal{N}$ the set of nonnegative integers.

2 The restricted modulo N extended gcd problem

In this section let F be any field. We will study the restricted modulo N extended gcd problem, i.e. finding for polynomials $a, b, N \in F[x]$, such that $N \neq 0$ and $\gcd(a, b, N) = 1$, a polynomial $f \in F[x]$ of minimal degree such that $\gcd(a + fb, N) = 1$. We will give a deterministic algorithm to solve this problem.

Definition 1 *For $N \in F[x] \setminus \{0\}$, let $g_F(N)$ be the minimal $t \in \mathcal{N}$ such that for all $a, b \in F[x]$ with $\gcd(a, b, N) = 1$, there exists an $f \in F[x]$ of degree $\leq t$ such that $\gcd(a + fb, N) = 1$.*

Lemma 1 *Let $a, b, N \in F[x]$, such that $N \neq 0$ and $\gcd(a, b, N) = 1$, and let $\{p_1, \ldots, p_k\}$ be the set of all irreducible monic divisors of N which do not divide b. Then, for $f \in F[x]$, we have that $\gcd(a + fb, N) = 1$ if and only if $\gcd(a + fb, p_i) = 1$ for $1 \leq i \leq k$.*

Proof. The 'only if' is clear. The 'if' follows from the fact that for p dividing both N and b we have that p does not divide $a + fb$ (since p does not divide a). ●

Lemma 2 *For all $N \in F[x] \setminus \{0\}$ we have $g_F(N) < \deg N$.*

Proof. Let $a, b \in F[x]$ such that $\gcd(a, b, N) = 1$, $\{p_1, \ldots, p_k\}$ the set of all irreducible monic divisors of N which do not divide b and $\tilde{N} = p_1 \cdots p_k$. Then, by Lemma 1, we have for all $f \in F[x]$ that $\gcd(a + fb, N) = 1$ if and only if $\gcd(a + fb, \tilde{N}) = 1$. Now let $f = (1 - a)/b \,(\mathrm{mod}\ \tilde{N})$. Then $a + fb \equiv 1 \,(\mathrm{mod}\ \tilde{N})$ so $\gcd(a + fb, \tilde{N}) = 1$ and furthermore $\deg f < \deg \tilde{N} \leq \deg N$. ●

Lemma 3 *Let $N \in F[x] \setminus \{0\}$ and l the number of irreducible monic divisors of N. If $S \subseteq F$ is such that $\#S > l$, then for all $a, b \in F[x]$, such that $\gcd(a, b, N) = 1$, there is an $s \in S$ such that $\gcd(a + sb, N) = 1$. In particular we have $g_F(N) = 0$ in that case.*

Proof. Let $a, b \in F[x]$, such that $\gcd(a, b, N) = 1$, and let $\{p_1, \ldots, p_k\}$ be the set of irreducible monic divisors of N which do not divide b. Since $k \leq l$, $\tilde{S} = S \setminus \{-a/b \,(\mathrm{mod}\ p_i) \mid 1 \leq i \leq k\}$ is not empty. From Lemma 1 it follows that for $s \in \tilde{S}$ we have $\gcd(a + sb, N) = 1$. ●

Notice that in practice we will not always know the number of irreducible monic divisors of N, but we can use the upper bound $\deg N$ for it.

Now we will give an algorithm for solving the restricted modulo N extended gcd problem. The algorithm is designed to solve efficiently a succession of these problems for a fixed N. In particular, we assume that we already have some factors of N. The algorithm then solves our problem and gives a possible refinement of the already known factors of N (which we remark need not be relatively prime). The refinement can then be used as input to future calls of the algorithm.

In the algorithm, M_u denotes the set $\{f \in F[x] \mid \deg f = u\}$ if $u \geq 0$ and $\{0\}$ if $u < 0$. An explanation of the various other sets in the algorithm is given in the proof of correctness following the algorithm.

```
Algorithm RGCD (Restricted modulo $N$ extended GCD problem)
Input: $a, b, N \in F[x]$, such that $N \neq 0$, $\deg a, \deg b < \deg N$ and $\gcd(a, b, N) = 1$, and $S \subseteq F[x]$ such that $(\prod_{s \in S} s) \mid N$ and $N$ divides a power of $\prod_{s \in S} s$.
Output: A minimal degree $f \in F[x]$ such that $\gcd(a + fb, N) = 1$. Also, $S' \subseteq F[x]$ such that $(\prod_{s' \in S'} s') \mid N$, $N$ divides a power of $\prod_{s' \in S'} s'$ and all $s \in S$ are a product of polynomials in $S'$.
if $\gcd(a, N) = 1$ then return $(0, S)$;
for $s \in S$ do $(a_s, b_s) := (a \,(\mathrm{mod}\ s), b \,(\mathrm{mod}\ s))$;
$S' := S$; $D := \emptyset$; $U := S$;
Comment: Extract factors in $S'$ relatively prime to $b$.
while $U \neq \emptyset$ do {
    Choose $u \in U$;
```

[1] Note that [9] contains an error. Fact 1 in Section 2 is an incorrect quote due to the author of [9]. A correct bound is $g(N) = O(r^2)$ (see [4]). Slight modifications of the algorithms in [9] admit the same asymptotic bounds for the running time and size of the output. A correction is in progress.

```
    $g_u := \gcd(b_u, u)$;
    if $\deg g_u = 0$ then $\{U := U \setminus \{u\};\ D := D \cup \{u\};\}$
    elif $\deg g_u = \deg u$ then $U := U \setminus \{u\}$;
    else {
        $S' := (S' \cup \{g_u, u/g_u\}) \setminus \{u\}$;
        $U := (U \cup \{u/g_u\}) \setminus \{u\}$;
        $(a_{u/g_u}, b_{u/g_u}) := (a_u \pmod{u/g_u}, b_u \pmod{u/g_u})$;}}
for $d \in D$ do $e_d := -a_d/b_d \pmod{d}$;
for $i = 0, 1, 2, \ldots$ do {
    if $\#F > \deg N$ then Choose $L \subseteq F$ such that $\#L = \deg N + 1$;
    else $L := M_i$;
    for $d \in D$ do
        Comment: Remove polynomials $f$ such that $a + fb \equiv 0 \pmod{d}$ from $L$.
        for $q \in M_{i-\deg d}$ do $L := L \setminus \{e_d + qd\}$;
    while $L \neq \emptyset$ do {
        Choose $f \in L$;
        for $d \in D$ do $f_d := f \pmod{d}$;
        for $d \in D$ do {
            $h_d := \gcd(a_d + f_d b_d, d)$;
            if $\deg h_d > 0$ then break;}
        if $\deg h_d = 0$ then
            return $(f, S')$;
        $e_{h_d} := e_d \pmod{h_d}$;
        $e_{d/h_d} := e_d \pmod{d/h_d}$;
        $S' := (S' \cup \{h_d, d/h_d\}) \setminus \{d\}$;
        $D := (D \cup \{h_d, d/h_d\}) \setminus \{d\}$;
        for $d \in \{h_d, d/h_d\}$ do
            for $q \in M_{i-\deg d}$ do $L := L \setminus \{e_d + qd\}$;}}
```

Theorem 1 *Algorithm RGCD is correct*

Proof. In the first while–loop of the algorithm we extract from the set S' the set D of polynomials which have no common divisor with b. When a factorization of a polynomial in S' is found during this process this is incorporated into S'.

In the big for–loop of the algorithm we try to find the wanted f by using a sieving process which excludes from the set of all possible candidates for f those polynomials which are easily computed to be bad.

By performing the sieving for polynomials of degree i for $i = 0, 1, 2, \ldots$ in succession we ensure that a minimal degree solution f will be returned. When, after performing the sieving process, we find a candidate f which yields a factorization of one of the polynomials in D we incorporate this in D and S'. Then we use the factors found to continue the sieving process.

Note that the following properties hold at all times during execution of the algorithm:

- S' has the right properties, i.e. $(\prod_{s' \in S'} s') \mid N$, N divides a power of $\prod_{s' \in S'} s'$ and all $s \in S$ are a product of polynomials in S'.
- $D \subseteq S'$, and for all $d \in D$ we have $\gcd(d, b) = 1$.
- The polynomials in $S' \setminus (D \cup U)$ divide b.

In particular, after execution of the first while-loop, the polynomials in $S' \setminus D$ divide b.

The while–loop of the sieving process is executed as long as we find factorizations of already known factors of N and there are candidates for f left in L. Since N has only finitely many factors the factor refining process must eventually stop. From this and the fact that L is finite, it follows that if the algorithm does not terminate, then i will eventually be increased.

It is clear that when f is excluded from the set L we have $\gcd(a + fb, N) \neq 1$. When $\#F \leq \deg N$ it follows from this, and the fact that $g_F(N)$ is finite, that the algorithm will eventually terminate. When $\#F > \deg N$ it follows from Lemma 3 that the algorithm will terminate.

If the algorithm returns f it is clear that $\gcd(a + fb, d) = 1$ for $d \in D$. Since $\gcd(a, b, N) = 1$ we also have $\gcd(a + fb, s') = 1$ for $s' \in S' \setminus D$. Since N divides a power of $\prod_{s' \in S'} s'$ this implies that $\gcd(a + fb, N) = 1$. •

In the following two theorems we bound the cost of one call to algorithm RGCD as well as the amortized cost when the algorithm is called a number of times for a fixed N. We denote by $P(n)$ a bound on the number of operations in F needed to perform the following operations in $F[x]$:

- Addition, subtraction, multiplication and division with remainder of polynomials of degree $< n$.
- Computing the extended gcd of two polynomials of degree $< n$, i.e. for polynomials f, g of degree $< n$, compute $g = \gcd(f, g)$ and $a, b \in F[x]$ of degree $< n$ such that $g = af + bg$.
- Computing the residues of a polynomial a modulo polynomials a_i where $\deg a, \sum \deg a_i < n$.

When $M(n)$ is a bound for the number of operations in F required to multiply two polynomials of degree $< n$, we have the following result.

Lemma 4 $P(n) = O(M(n) \log(n))$.

Proof. See [1]. •

We assume that there is a monotonically nondecreasing function $f(n)$ such that $P(n) = nf(n)$. In practice this will be no problem. We then have the following lemma.

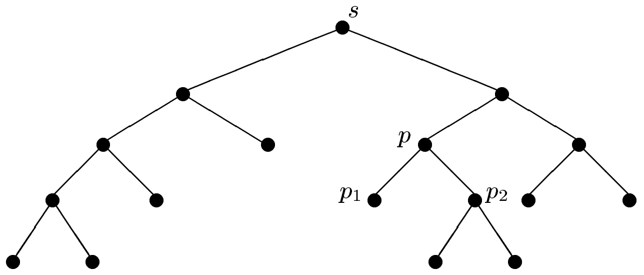
Lemma 5 *For positive integers $n, n_1, \ldots, n_l$ such that $n = \sum_{i=1}^{l} n_i$ we have $\sum_{i=1}^{l} P(n_i) \leq P(n)$.*

Proof. $\sum_{i=1}^{l} P(n_i) = \sum_{i=1}^{l} n_i f(n_i) \leq \sum_{i=1}^{l} n_i f(n) = nf(n) = P(n)$. •

First we will bound the cost of one call to RGCD.

Theorem 2 *Let l be the number of times a factorization of one of the (up to then) known divisors of N is found during execution of algorithm RGCD. Then the algorithm takes $O((l+1)P(\deg N) + (\#F)^{g_F(N)} P(g_F(N)) \deg N)$ operations in F.*

Proof. We can view the evolution of factorization of one of the polynomials $s \in S$ as a binary tree B_s, e.g.



The root of B_s is labeled by s. If a vertex is labeled by some polynomial p which during execution of the algorithm is factored as $p = p_1 p_2$, then the "children" of p will be labeled by p_1 and p_2. In the sequel we will identify a vertex with its label. Let V denote the set of all vertices in all B_s ($s \in S$), V_r the set of roots in V, V_l the set of leaves in V and V_p the set of parents in V, i.e. $V_p = V \setminus V_l$. It is easy to see that $\#V_p < \#V_l$. Furthermore $\#V_l$ is less than the number of irreducible factors of N, so $\#V_l \leq \deg N$. A factorization of one of the (up to then) known factors of N corresponds to a vertex $v \in V_p$. So the number l in the theorem equals $\#V_p$. It is easy to prove the identity $\#(V \setminus V_r) = 2\#V_p$.

The cost of computing $\gcd(a, N)$ is $O(P(\deg N))$. Computing the a_s and b_s for all s takes $O(P(\deg N))$ since $\sum_{s \in S} \deg s \leq \deg N$.

We have to compute the eight quantities g_u, u/g_u, a_{u/g_u}, b_{u/g_u}, e_d, d/h_d, e_{h_d} and e_{d/h_d} all for particular subsets of V. The cost of these computations can be bounded by $8(\sum_{r \in V_r} P(\deg r) + \sum_{v \in V \setminus V_r} P(\deg N)) \leq 8(2l+1)P(\deg N)$, since $\sum_{r \in V_r} \deg r \leq \deg N$.

The computation of the f_d for all $d \in D$ can be done in $O(P(\deg N))$ since $\deg f < \deg N$ and $\sum_{d \in D} \deg d \leq \deg N$. We have to perform this computation at most $l+1$ times.

The computation of the h_d for all $d \in D$ can be done in $\sum_{d \in D} P(\deg d) \leq P(\deg N)$. We have to perform this computation at most $l+1$ times.

Finally we have to compute the cost of excluding the $e_d + qd$ from L. This has to be done for all d in a subset of V. When $\#F > \deg N$, we only have to exclude the e_d from L; the number of these is bounded by $O(\#V) = O(\deg N)$. Now suppose that $\#F \leq \deg N$. When $\deg d > i$ we only have to exclude e_d from L in the case that $\deg e_d = i$. When $\deg d \leq i$ we have $\#M_{i-\deg d} = (\#F - 1)(\#F)^{i-\deg d}$. Summing over all i, we see that for a particular d, the total number of times we have to exclude a polynomial from an L is ≤ 1 when $\deg d > g_F(N)$ and

$$\leq 1 + (\#F - 1) \sum_{j=0}^{g_F(N)-\deg d} (\#F)^j = (\#F)^{g_F(N)-\deg d+1},$$

when $\deg d \leq g_F(N)$. Summing over all $d \in V$, we see that the total number of times we have to exclude a polynomial from an L is

$$\begin{aligned}
\leq & \sum_{\substack{d \in V \\ \deg d \leq g_F(N)}} (\#F)^{g_F(N)-\deg d+1} + \sum_{\substack{d \in V \\ \deg d > g_F(N)}} 1 \\
\leq & \; 2\deg N + (\#F)^{g_F(N)+1} \sum_{\substack{d \in V \\ \deg d \leq g_F(N)}} (\#F)^{-\deg d} \\
\leq & \; 2\deg N + (\#F)^{g_F(N)+1} \sum_{\substack{d \in V \\ \deg d \leq g_F(N)}} (\#F)^{-1} \\
\leq & \; 2\deg N + (\#F)^{g_F(N)+1}(2\deg N/\#F) \\
= & \; O((\#F)^{g_F(N)} \deg N)
\end{aligned}$$

The cost of computing $e_d + qd$ for $q \in M_{i-\deg d}$ is $O(P(i)) = O(P(g_F(N)))$. Summing up all costs proves the theorem. ●

Remark 1 *In the computation of the cost of RGCD we have not taken into account the overhead caused by handling the set L. In practice L can be implemented by an array. Exclusion of a polynomial from L is then done by zeroing an entry in L. When F is the field with p^d elements, the computation of the index of a polynomial g in L can be done by evaluating a polynomial of degree* $\deg g$ *with coefficients $< p^d$ in p^d. We assume that the cost for this is equivalent to $O(\deg g)$ operations in F. Since* $\deg g \leq g_F(N)$ *this can be neglected in our cost computation.*

Furthermore, when keeping track of the position at which we are searching for $f \in L$, the total cost of this search is linear in the total length of all arrays. When $\#F > \deg N$ this length is $1 + \deg N$, when $\#F \leq \deg N$ the length is $(\#F)^{g_F(N)+1} \leq (\#F)^{g_F(N)} \deg N$.

Next we will bound the cost of solving a succession of restricted modulo N extended gcd problems for a fixed modulus N.

Theorem 3 *Using the algorithm RGCD, computing (in succession for $i = 1, \ldots, s$) for $a_i, b_i \in F[x]$, such that* $\deg a_i, \deg b_i < \deg N$ *and* $\gcd(a_i, b_i, N) = 1$*, $f_i \in F[x]$ of minimal degree such that* $\gcd(a_i + f_i b_i, N) = 1$ *can be done in* $O(P(\deg N)\deg N + s(P(\deg N) + (\#F)^{g_F(N)} P(g_F(N)) \deg N))$ *operations in F.*

Proof. We have to perform RGCD s times. For the first call to RGCD we can use $S = \{N\}$. By using finer factorizations found by RGCD in future calls we see that the total number of factorizations of known factors of N is bounded by $\deg N$. ●

3 The function $g_F(N)$

In this section we will study the function $g_F(N)$ more closely. In fact we will bound $g_F(N)$ for general N by $g_F(D)$ for some special D (D depending on N). This will be based on a conjectural bound for $g_F(D)$ which then can be used to bound general $g_F(N)$'s. Unfortunately we are not able to prove the conjecture, but we will give some evidence to support it.

Lemma 6 *For $N \in F[x] \setminus \{0\}$, $g_F(N)$ is the minimal $t \in \mathcal{N}$ such that for all $a \in F[x]$ there exists an $f \in F[x]$ of degree $\leq t$ such that* $\gcd(a - f, N) = 1$.

Proof. Let $h_F(N)$ be the minimal $t \in \mathcal{N}$ such that for all $a \in F[x]$ there exists an $f \in F[x]$ of degree $\leq t$ such that $\gcd(a - f, N) = 1$. Taking $b = -1$ we see that $h_F(N) \leq g_F(N)$. Now let $a, b \in F[x]$ such that $\gcd(a, b, N) = 1$, $\tilde{N}$ the product of all irreducible monic divisors of N which do not divide b and $\tilde{a} \equiv -a/b \pmod{\tilde{N}}$. Let $f \in F[x]$ of degree $\leq h_F(N)$ such that $\gcd(\tilde{a} - f, N) = 1$. Then also $\gcd(\tilde{a} - f, \tilde{N}) = 1$ and since $\gcd(b, \tilde{N}) = 1$ also $\gcd(-b(\tilde{a} - f), \tilde{N}) = 1$. But $-b(\tilde{a} - f) \equiv a + fb \pmod{\tilde{N}}$ so $\gcd(a + fb, \tilde{N}) = 1$ and so, by Lemma 1, $\gcd(a + fb, N) = 1$. From this it follows that $g_F(N) \leq h_F(N)$. ●

We see that the function g_F is similar to the well-known Jacobsthal function g from number theory (see [5], [8]) which can be defined by $g(n) = \min\{t \in \mathcal{N} \mid \forall_{a \in \mathcal{N}} \exists_{i \in \{1,\ldots,t\}} \colon \gcd(a + i, n) = 1\}$, for $n \in \mathcal{N}$. We therefore call g_F the *polynomial Jacobsthal function for F*. In [4] it is proven that $g(n) = O(k^2)$, where k is the number of prime divisors of n.

From now on let $N \in F[x]\setminus\{0\}$. Let $I(N)$ be the set of irreducible monic divisors of N. We have already seen (Lemma 3) that when F contains more than $\#I(N)$ elements, then $g_F(N) = 0$. So when F is an infinite field we have $g_F(N) = 0$ for all N.

So from now on assume that F is a finite field with q elements. First we will introduce some notation. For $f \in F[x]$ we denote by d_f the degree of f. Furthermore, let P be the set of all irreducible monic polynomials over F and for $t \in \mathcal{N}$ let $\overline{M}_t = \{f \in F[x] \mid d_f \leq t\}$. For $t \in \mathcal{N}$, $p \in P$ and $a \in F[x]$ let

$$R^t_{p,a} = \{f \in \overline{M}_t \mid f \equiv a \pmod p\}.$$

So $R^t_{p,a}$ is the set of all polynomials f of degree $\leq t$ such that $\gcd(a-f,p) \neq 1$. For $f \in \overline{M}_t$ we have $\gcd(a-f,p) = 1$ if and only if $f \notin R^t_{p,a}$ whence $\gcd(a-f,N) = 1$ if and only if $f \notin \cup_{p\in I(N)} R^t_{p,a}$. For $t \in \mathcal{N}$ and $U \subseteq P$ let

$$N^t_U = \max\{\#(\cup_{p\in U} R^t_{p,a}) \mid a \in F[x]\}.$$

We see that for $t \in \mathcal{N}$, there exists for all $a \in F[x]$ an $f \in \overline{M}_t$ such that $\gcd(a-f,N) = 1$ if and only if $N^t_{I(N)} < \#\overline{M}_t = q^{t+1}$. We get the following lemma.

Lemma 7 *$g_F(N)$ is the minimal $t \in \mathcal{N}$ such that $N^t_{I(N)} < q^{t+1}$.*

The following lemma states that we can replace irreducible polynomials of high degree by other irreducible polynomials.

Lemma 8 *Let $t \in \mathcal{N}$, $U \subseteq P$ and $k, l \in P \setminus U$ with $d_k \geq t+1$. Then $N^t_{U\cup\{k\}} \leq N^t_{U\cup\{l\}}$.*

Proof. When $N^t_U = q^{t+1}$, then $N^t_{U\cup\{k\}} = N^t_{U\cup\{l\}} = q^{t+1}$. So assume that $N^t_U < q^{t+1}$. Let $a \in F[x]$ such that

$$N^t_{U\cup\{k\}} = \#((\cup_{p\in U} R^t_{p,a}) \cup R^t_{k,a}).$$

Now

$$\begin{aligned}\#((\cup_{p\in U} R^t_{p,a}) \cup R^t_{k,a}) &\leq \#(\cup_{p\in U} R^t_{p,a}) + \#R^t_{k,a} \\ &\leq \#(\cup_{p\in U} R^t_{p,a}) + 1 \\ &\qquad \text{(since } d_k \geq t+1\text{)} \\ &\leq N^t_U + 1,\end{aligned}$$

so $N^t_U \geq N^t_{U\cup\{k\}} - 1$. Let $\tilde{a} \in F[x]$ such that

$$N^t_U = \#(\cup_{p\in U} R^t_{p,\tilde{a}}).$$

Since $N^t_U < q^{t+1}$ there exists an $f \in \overline{M}_t \setminus (\cup_{p\in U} R^t_{p,\tilde{a}})$. Let $\hat{a} \in F[x]$ such that $\hat{a} \equiv \tilde{a} \pmod p$ for all $p \in U$ and $\hat{a} \equiv f$ (mod l). Then $\cup_{p\in U} R^t_{p,\hat{a}} = \cup_{p\in U} R^t_{p,\tilde{a}}$ and thus

$$\begin{aligned}N^t_{U\cup\{l\}} &\geq \#((\cup_{p\in U} R^t_{p,\hat{a}}) \cup R^t_{l,\hat{a}}) \\ &> \#(\cup_{p\in U} R^t_{p,\hat{a}}) \\ &= N^t_U \\ &\geq N^t_{U\cup\{k\}} - 1.\end{aligned}$$

●

Using Lemma 8 a number of times, we can prove the following lemma, where for a positive integer t we define $I_t = P \cap \overline{M}_t$.

Lemma 9 *Let t be a positive integer and $U \subseteq P$ such that $\#U \leq \#I_t$. Then $N^t_U \leq N^t_{I_t}$.*

Proof. Let $\tilde{U} = U \setminus I_t$. If $\tilde{U} = \emptyset$, then $U \subseteq I_t$ and the lemma follows. If $\tilde{U} \neq \emptyset$, let $k \in \tilde{U}$ and $l \in I_t \setminus U$. By Lemma 8 we know that $N^t_U \leq N^t_{(U\cup\{l\})\setminus\{k\}}$ and the theorem now follows by induction on $\#\tilde{U}$. ●

To Lemma 7 and Lemma 9 we get the following corollary, where for a positive integer t we define $D_t = \prod_{p\in I_t} p$, i.e. D_t is the product of all irreducible monic polynomials of degree $\leq t$. The corollary states that, in some sense, D_t is the worst polynomial we can have with a certain bounded number of irreducible divisors.

Corollary 1 *Let $N \in F[x]$ such that $\#I(N) \leq \#I_t$. Then $g_F(N) \leq g_F(D_t)$.*

We conjecture the following bound for the value of $g_F(D_t)$.

Conjecture 1 *For all positive integers t we have $N^t_{I_t} < q^{t+1}$, i.e. $g_F(D_t) \leq t$.*

Unfortunately we are not able to prove the conjecture. We have however some evidence to support it. First of all the conjecture is true for $t = 1$.

Theorem 4 *$N^1_{I_1} < q^2$.*

Proof. Let $a \in F[x]$ such that $N^1_{I_1} = \#(\cup_{p\in I_1} R^1_{p,a})$. Let $b \in F\setminus\{a(0)\}$, $S = \{(a(i)-b)/i \mid i \in F\setminus\{0\}\}$ and $c \in F\setminus S$. Then $b + cx \in \overline{M}_1 \setminus (\cup_{p\in I_1} R^1_{p,a})$. ●

Furthermore exhaustive tests have proven the following fact.

Fact 1 *For $(q,t) \in \{(2,2),(2,3),(2,4),(3,2),(4,2)\}$ we have $N^t_{I_t} < q^{t+1}$*

Also numerous (though not exhaustive) experiments indicate that $N^t_{I_t} = \#(\cup_{p\in I_t} R^t_{p,a})$ for all $a \in F[x]$ of degree $\leq t$, which means that $a \in F[x]$ of degree $\leq t$ is the worst case we can get. For $a \in F[x]$ with degree $\leq t$ we have $\overline{M}_t \setminus (\cup_{p\in I_t} R^t_{p,a}) = \{a + c \mid c \in F \setminus \{0\}\}$ so $\#(\cup_{p\in I_t} R^t_{p,a}) = q^{t+1} - (q-1) < q^{t+1}$.

Some of the upcoming results will depend on Conjecture 1. These results will be labeled "Conjecture Corollary".

Conjecture Corollary 1 *Let t be a positive integer. Then $g_F(D_t) = t$.*

Proof. Taking $a = x^t$ we have that $\gcd(a - f, D_t) \neq 1$ for all $f \in \overline{M}_t \setminus \{x^t - c \mid c \in F\}$. This shows that $g_F(D_t) \geq t$. The result now follows from Conjecture 1 ●

Conjecture Corollary 2 *$d_N > \sum_{p\in I_{g_F(N)-1}} d_p$.*

Proof. From Corollary 1 and Conjecture 1 we get

$$\#I_{g_F(N)-1} < \#I(N).$$

Since I_t contains only irreducible polynomials of smallest possible degree the corollary easily follows. ●

The following lemma follows from some elementary algebra.

Lemma 10 $\sum_{p \in I_t} d_p \geq q^t$.

Proof. Let K be an extension field of degree t over F. For $a \in K$ let m_a be the minimal polynomial of a over F. We say that $a, b \in K$ are equivalent ($a \sim b$) if $m_a = m_b$. This defines an equivalence relation on K. Let $K/\!\sim$ denote the set of equivalence classes and for $c \in K/\!\sim$ let m_c be the corresponding minimal polynomial. Since $F \subseteq K$ is a Galois extension the number of elements in the equivalence class of $a \in K$ is equal to the degree of m_a. From this we get

$$\sum_{p \in I_t} d_p \geq \sum_{c \in K/\sim} d_{m_c} = \sum_{c \in K/\sim} \#c = \#K = q^t.$$

●

From Lemma 10 and Conjecture Corollary 2 we now get the following corollary.

Conjecture Corollary 3 $d_N > q^{g_F(N)-1}$.

3.1 Average case analysis: a partial result

In this section we will show that for many polynomials $a \in F[x]$ there exists a constant $c \in F$ such that $\gcd(a - c, N) = 1$. From this it follows that the running time of algorithm RGCD will in general be much better than the worst case bound given by Theorem 2. We emphasize, though, that the result presented here is still far from a complete average case analysis. We will omit the proofs of the various statements.

It is clear that we only have to consider $a \in \overline{M}_{d_N-1}$, since we can take a modulo N. The following lemma gives a formula for the portion of polynomials $a \in \overline{M}_{d_N-1}$ for which there exists a $c \in F$ such that $\gcd(a - c, N) = 1$.

Lemma 11 *Let $N \in F[x] \setminus \{0\}$, $I(N) = \{p_1, \ldots, p_s\}$ and $N = p_1^{e_1} \cdots p_s^{e_s}$. Then*

$$\#\{a \in \overline{M}_{d_N-1} \mid \exists_{c \in F}\colon \gcd(a-c, N) = 1\} / \#\overline{M}_{d_N-1}$$
$$= \sum_{j=1}^{q} (-1)^{j-1} \binom{q}{j} \prod_{i=1}^{s} (1 - j/q^{d_{p_i}}). \qquad (2)$$

We won't give extensive estimates of the right hand side of (2) but will only give some examples. By performing some manipulations we see that the right hand side of (2) is equal to 1 when $s < q$ (which agrees with Lemma 3) and can be bounded from below by $1 - q!/q^q$ when $s = q$, which converges to 1 very rapidly. Already for $q \geq 9$ we have $1 - q!/q^q > 0.999$. When we take for N the product of all monic irreducible polynomials of degree ≤ 2 we get the values of the following table for the right hand side of (2).

q	
2	0.375
3	0.572
5	0.780
8	0.914
23	0.999

4 The generalized modulo N extended gcd problem

For a matrix A over $F[x]$ we write $\gcd(A)$ to mean the gcd of all entries in A. The generalized modulo N extended gcd problem takes as input an $A \in F[x]^{n \times m}$ together with a nonzero $N \in F[x]$, and asks for a $c \in F[x]^{m \times 1}$ such that $\gcd(Ac, N) = \gcd(A, N)$. Here we are interested in computing a solution c with small degree entries, that is, with degrees of entries bounded by $g_F(N)$. We first give a solution to this problem for the case $m = 2$. In what follows we write A_{*i} to denote the i-th column vector of A and c_i to denote the i-th entry of c.

Lemma 12 *There exists an algorithm that returns a solution $f \in F[x]$ to the equation $\gcd(A_{*1} + fA_{*2}, N) = \gcd(A, N)$ where $A \in F[x]^{n \times 2}$. The solution will satisfy $f \leq g_F(N)$. If degrees of entries in A are bounded by $\deg N$, then the cost of the algorithm is $O(nP(\deg N))$ field operations plus the cost of computing a single solution to the restricted modulo N extended gcd problem.*

Proof. Let T be a copy of A. At a cost of $O(nP(\deg N))$ field operations transform T, using unimodular row operations and reduction modulo N, to have the form

$$T = \begin{bmatrix} a & b \\ * & 0 \\ * & 0 \\ \vdots & \vdots \\ * & 0 \end{bmatrix},$$

where $*$ denotes a possibly nonzero entry and $b = \gcd(A_{*2})$. Compute f to be a solution to the restricted modulo N extended gcd problem $\gcd((a/g) + f(b/g), N) = 1$ where $g = \gcd(a, b)$. Then $\gcd(a + fb, N) = \gcd(a, b, N)$ and $\deg f \leq g_F(N)$. It follows that $\gcd(T_{*1} + fT_{*2}, N) = \gcd(T, N)$. Since T is left equivalent to A (modulo N) we must also have $\gcd(A_{*1} + fA_{*2}, N) = \gcd(A, N)$. ●

Theorem 5 *There exists an algorithm that returns a solution $c \in F[x]^{m \times 1}$ to the generalized modulo N extended gcd problem with input $A \in F[x]^{n \times m}$. The solution will satisfy $c_1 = 1$ and $\deg c_i \leq g_F(N)$ for $2 \leq i \leq n$. If degrees of entries in A are bounded by $\deg N$, then the cost of the algorithm is $O(nmP(\deg N))$ field operations plus the cost of computing $m - 1$ solutions to the restricted modulo N extended gcd problem.*

Proof. Set $c_1 \leftarrow 1$. Initialize B to be a copy of A_{*1}. For $i = 2, 3, \ldots, m$ perform the following steps. (1) Using the algorithm of Lemma 12 compute an f such that $\gcd(B + fA_{*i}, N) = \gcd(B, A_{*i}, N)$. (2) Add f times A_{*i} to B and reduce modulo N the entries in B. (3) Set $c_i \leftarrow f$. ●

5 An algorithm for the Smith normal form

Given a nonsingular polynomial input matrix $A \in F[x]^{n \times n}$, we want to produce unimodular transforming matrices $U, V \in F[x]^{n \times n}$ such that $S = UAV = \operatorname{diag}(s_1, s_2, \ldots, s_n)$ is the Smith normal form of A. In [9] we presented an algorithm for producing transforming matrices in the case of an integer input matrix. That algorithm depends on a subroutine for solving the modulo N extended gcd problem for integers and incorporates ideas from [3, 7, 12]. Using the subroutine for solving the generalized modulo N extended gcd problem for polynomials presented in section 4, the algorithm in [9] is easily adapted to work for a polynomial input matrix.

The approach taken in [9], first used in [7], is to compute a unit lower triangular matrix C such that AC can be transformed using only unimodular row operations to an upper triangular T with i-th diagonal entry s_i for $1 \le i \le n$. We call C a *Smith conditioner* for A. In [12] T is called a *triangular Smith form* of AC. Once a Smith conditioner and triangular Smith form have been computed, transforming matrices are easily recovered.

In the Las Vegas probabilistic algorithm in [7], choosing the entries in C randomly from a subset of F of cardinality $4n^3d$ will ensure probability of failure less than 1/2. The algorithm in [12] computes $C \in F^{n\times n}$ deterministically but also requires that F has at least $2nd$ distinct elements. If $\#F$ is too small then the algorithms in [7, 12] may require computing over an algebraic extension of F which may lead to algebraic numbers in the transforming matrices. In the following example we give an input matrix for which there does not exist a Smith conditioner over F.

Example 1 *Let F be the field of two elements and consider the nonsingular input matrix*

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

The gcd of all entries of A is 1. We want to condition A so that the gcd of the entries in the first column is equal 1. When we multiply A on the right with

$$C = \begin{bmatrix} 1 & 0 \\ c & 1 \end{bmatrix}$$

the gcd of the entries of the first column is for $c = 0$ equal to x and for $c = 1$ equal to $x + 1$. In both cases this is not the gcd of all entries of A. Now let $F \subseteq K$ be an extension of degree 2 and let $y \in K$ such that $y^2 + y + 1 = 0$. Taking $c = y$ we get

$$AC = \begin{bmatrix} x+y & 1 \\ (y+1)x^2+yx+y & x^2+x+1 \end{bmatrix},$$

and $\gcd(x+y, (y+1)x^2+yx+y) = 1 = (1+x)(x+y) + y((y+1)x^2+yx+y)$. *This gives*

$$\begin{bmatrix} x+1 & y \\ (y+1)x^2+yx+y & x+y \end{bmatrix} AC =$$

$$\begin{bmatrix} 1 & yx^2+(y+1)x+y+1 \\ 0 & x^3+x \end{bmatrix}.$$

By clearing the top-right entry by a column operation we finally get

$$UAV = \begin{bmatrix} 1 & 0 \\ 0 & x^3+x \end{bmatrix},$$

where

$$U = \begin{bmatrix} x+1 & y \\ (y+1)x^2+yx+y & x+y \end{bmatrix},$$

$$V = \begin{bmatrix} 1 & yx^2+(y+1)x+y+1 \\ y & (y+1)x^2+x \end{bmatrix}.$$

We see that both U and V contain algebraic numbers.

We can avoid the use of algebraic numbers by allowing polynomials for the entries of the Smith conditioner matrix C.

Example 2 *Consider the same input matrix as in Example 1. Taking $c = x + 1$ we get*

$$AC = \begin{bmatrix} 1 & 1 \\ x^3+x^2+1 & x^2+x+1 \end{bmatrix},$$

and $\gcd(1, x^3+x^2+1) = 1 = 1 \cdot 1 + 0 \cdot (x^3+x^2+1)$. *This gives*

$$\begin{bmatrix} 1 & 0 \\ x^3+x^2+1 & 1 \end{bmatrix} AC = \begin{bmatrix} 1 & 1 \\ 0 & x^3+x \end{bmatrix},$$

and by clearing the top-right entry by a column operation we get

$$UAV = \begin{bmatrix} 1 & 0 \\ 0 & x^3+x \end{bmatrix},$$

where

$$U = \begin{bmatrix} 1 & 0 \\ x^3+x^2+1 & 1 \end{bmatrix}, V = \begin{bmatrix} 1 & 1 \\ x+1 & x \end{bmatrix}.$$

Now both U and V don't contain algebraic numbers.

We now adapt the algorithm in [9] to work for polynomial input matrices. The algorithm computes the columns of C as solutions to instances of the generalized modulo N extended gcd problem with $N = \det A$. This leads to the following two theorems.

Theorem 6 *There exists a deterministic algorithm that takes as input a nonsingular $A \in F[x]^{n\times n}$, and returns as output a Smith conditioner C for A together with a triangular Smith normal form T of AC which has degrees of entries in column i bounded by $\deg s_i$. If degrees of entries in A are bounded by d, then the cost of the algorithm is $O(n^3 P(nd))$ field operations plus the cost of solving $n-1$ instances of the generalized modulo N extended gcd problem with $N = \det A$ and with dimension bounded by n.*

Proof. The algorithm is analogous to that for integer matrices presented in [9]. For a detailed presentation and a proof of correctness see [9]. ●

Theorem 7 *Let $A \in F[x]^{n\times n}$ be nonsingular with degrees of entries bounded by d. Let C be a Smith conditioner of A and T a triangular Smith form of AC with degrees of entries in column i bounded by $\deg s_i$. If degrees of entries in C are bounded by d_C, then unimodular U and V such that $UAV = S$ can be recovered from T and C in $O(n^3 P(n(d + d_C)))$ field operations. Degrees of entries in U will be bounded by $(n-1)(d+d_C)$ and degrees of entries in column j of V will be bounded by $d_C + \deg s_j$.*

Proof. Note that S is given by the diagonal entries of T. Compute $V \leftarrow C(S^{-1}T)^{-1}$, and $U \leftarrow (TC^{-1}A^{\mathsf{adj}})(1/\det(A))$. Then $UAV = S$ as required. It remains to establish the degree bounds for entries in U and V. The matrix $S^{-1}T$ will be unit upper triangular with the degree of entry in column j row i bounded by $\deg s_j - \deg s_i$. The inverse matrix $R = (S^{-1}T)^{-1}$ will also be unit upper triangular with the entry in column j row i bounded by $\deg s_j - \deg s_i$ [12, Proof of Corollary 4.1]. The claimed bounds for the degrees of entries in V follow easily. Entries in A^{adj} and C^{-1} will be bounded in degree by $(n-1)d$ and $(n-1)d_C$ respectively. Noting that degrees of entries in T are bounded by $\deg(\det A)$ we get the bound $(n-1)(d+d_C)$ for degrees of entries in U. The cost of recovering U and V follows from the degree bounds. ●

From Conjecture Corollary 3 and Theorems 3, 5, 6 and 7 we get the following conjectured corollary.

Conjecture Corollary 4 *Let F be a finite field and $q = \#F$. There exists a deterministic algorithm that takes as input a nonsingular $A \in F[x]^{n\times n}$ with degrees of entries bounded by d, and returns as output the Smith normal form S of A together with unimodular transforming matrices U and V such that $UAV = S$. Degrees of entries in U will be bounded by $(n-1)(d + \lceil \log_q nd \rceil)$ and degrees of entries in column j of V will be bounded by $\lceil \log_q nd \rceil + \deg s_j$ for $1 \le j \le n$. The cost of the algorithm is $O(n^3 P(n(d + \log n)) + ndP(nd) + n^4 d^2 qP(\log nd))$ operations from F.*

As a corollary to Lemma 3 and Theorems 3, 5, 6 and 7 we get the following.

Theorem 8 *Let $\#F > nd$. There exists a deterministic algorithm that takes as input a nonsingular $A \in F[x]^{n\times n}$ with degrees of entries bounded by d, and returns as output the Smith normal form S of A together with unimodular transforming matrices U and V such that $UAV = S$. The degrees of entries in U will be bounded by $(n-1)d$. The degrees of entries in column j of V will be bounded by $\deg s_j$ for $1 \le j \le n$. The cost of the algorithm is $O(n^3 P(nd) + ndP(nd))$ operations from F.*

6 Conclusions and future work

In this paper we have presented a deterministic algorithm to solve the restricted modulo N extended gcd problem. In the large field case this algorithm always gives a solution of degree 0. In the small field case we have, besides a trivial upper bound, conjectured an optimal bound for the degree of a solution. We have given some evidence to support the conjecture but we are still far from a proof.

Furthermore we have shown how this algorithm can be used to adapt the algorithm in [9] to compute transforming matrices for the Smith form of a nonsingular matrix over $F[x]$. This algorithm always returns transforming matrices over $F[x]$ and gives good bounds on the degrees of the entries in the transforming matrices (these bounds depend however on the conjecture in the small field case). Previous algorithms in [7, 12] may return transforming matrices with entries in some algebraic extension field of F.

In the small field case the worst case complexity of our algorithms depend on the conjecture. It is not difficult to compute a solution to the restricted modulo N extended gcd problem of not necessarily minimal but moderate degree (see [11]). Using this, our algorithm for Smith form computation will become independent of the conjecture although the degree bounds will be less attractive. We will not go into detail.

In our algorithm, following the approach in [2], we perform a big part of the computations modulo $\det A$. This ensures a good bound on the degrees of intermediate polynomials. The algorithm in [12] computes transforming matrices for the Smith form of a matrix over $\mathcal{Q}[x]$, where $\mathcal{Q}$ is the field of rational numbers. It is designed to give good bounds on the size of intermediate and final rational numbers.

In the future we will present an algorithm for Smith form over $\mathcal{Q}[x]$ which is based on homomorphic imaging; a key step in this algorithm will be to compute transforming matrices for the Smith form over $F[x]$, where F is a prime field, using the algorithm we have presented here. Also, the Smith form algorithm we have presented here should be generalized to work for rectangular and/or singular input matrices.

Acknowledgment We are grateful to the referees for their useful comments.

References

[1] Aho, A. V., Hopcroft, J. E., and Ullman, J. D. *The Design and Analysis of Computer Algorithms.* Addison-Wesley, 1974.

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

[3] Giesbrecht, M. Fast computation of the Smith normal form of an integer matrix. In *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC '95* (1995), A. H. M. Levelt, Ed., pp. 110–118.

[4] Iwaniec, H. On the problem of Jacobsthal. *Demonstratio Mathematica 11*, 1 (1978), 225—231.

[5] Jacobsthal, E. Über Sequenzen ganzer Zahlen, von denen keine zu n teilerfremd ist I–III. *Norske Vid. Selsk. Forhdl. 33* (1960), 117–124, 125–131, 132–139.

[6] Kaltofen, E., Krishnamoorthy, M. S., and Saunders, B. D. Fast parallel computation of Hermite and Smith forms of polynomial matrices. *SIAM Journal of Algebraic and Discrete Methods 8* (1987), 683–690.

[7] Kaltofen, E., Krishnamoorthy, M. S., and Saunders, B. D. Parallel algorithms for matrix normal forms. *Linear Algebra and its Applications 136* (1990), 189–208.

[8] Kanold, H.-J. Über eine zahlentheoretische Funktion von Jacobsthal. *Math. Annalen 170* (1967), 314–326.

[9] Storjohann, A. A solution to the extended gcd problem with applications. In *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC '97* (1997).

[10] Storjohann, A., and Labahn, G. A fast Las Vegas algorithm for computing the Smith normal form of a polynomial matrix. *Linear Algebra and its Applications 253* (1997), 155—173.

[11] Storjohann, A., and Mulders, T. Fast algorithms for linear algebra modulo N. To appear in *Proc. of Sixth Ann. Europ. Symp. on Algorithms: ESA'98.*

[12] Villard, G. Generalized subresultants for computing the Smith normal form of polynomial matrices. *Journal of Symbolic Computation 20*, 3 (1995), 269—286.

[13] Villard, G. Fast parallel algorithms for matrix reduction to normal forms. *Applicable Algebra in Engineering, Communication and Control 8* (1997), 511—537.