Fast Jacobian Group Arithmetic on C_{ab} Curves

Ryuichi Harasawa[1] and Joe Suzuki[1]

Department of Mathematics, Graduate School of Science, Osaka University,
1-1 Machikaneyama, Toyonaka, Osaka 560-0043, Japan
{harasawa, suzuki}@math.sci.osaka-u.ac.jp

Abstract

The goal of this paper is to describe a practical and efficient algorithm for computing in the Jacobian of a large class of algebraic curves over a finite field. For elliptic and hyperelliptic curves, there exists an algorithm for performing Jacobian group arithmetic in $O(g^2)$ operations in the base field, where g is the genus of a curve. The main problem in this paper is whether there exists a method to perform the arithmetic in more general curves. Galbraith, Paulus, and Smart proposed an algorithm to complete the arithmetic in $O(g^2)$ operations in the base field for the so-called superelliptic curves. We generalize the algorithm to the class of C_{ab} curves, which includes superelliptic curves as a special case. Furthermore, in the case of C_{ab} curves, we show that the proposed algorithm is not just general but more efficient than the previous algorithm as a parameter a in C_{ab} curves grows large.

Keywords: discrete logarithm problem, algebraic curve cryptography, Jacobian group, ideal class group, superelliptic curves, C_{ab} curves

1 Introduction

This paper is motivated by cryptography based on the intractability of the discrete logarithm problem (DLP) in the divisor class group of a curve. While elliptic curve cryptography has drawn considerable public attention in recent years, cryptosystems using hyperelliptic curves are currently getting accepted as well, which seems to be based on the following considerations:

1. the order of a Jacobian group can be large compared to the size of the field if the genus g of the curve is large (the Hasse-Weil bound [15]);
2. a novel method for solving the elliptic curve DLP that would be proposed in the future may not be applied to non-elliptic curves; and
3. recently, several fast algorithms for performing arithmetic on hyperelliptic curves have been proposed.

For elliptic curves, a method for performing addition among Jacobians has been known from a long ago, and its group arithmetic is given as a simple formula [13]. On the other hand, an efficient method of Jacobian group arithmetic for hyperelliptic curves has been given by D.G. Cantor [2]. (Although Cantor assumed the characteristic is not two, N. Koblitz recently excluded the constraint [7].) The only problem in addition of divisor classes is to compute good prescribed representatives of a class. In the case of hyperelliptic curves, following Cantor [2] several methods for this have been proposed (see N. Smart [14] for details), and the algorithms realized in $O(g^2)$ operations in the base field are supposed to be the most efficient methods thus far.

In this paper, we address the problem whether or not there exists a method for performing Jacobian group arithmetic in $O(g^2)$ operations in the base field for more general curves than elliptic and hyperelliptic curves.

This problem has been solved in the affirmative for a class of curves called superelliptic curves (Galbraith, Paulus, and Smart [5]):

$$C/F_q : Y^a = \sum_{i=0}^{b} \alpha_i X^i \ ,$$

where $\alpha_i \in F_q$, $\alpha_b \neq 0$, a and b are coprime, and the curve is assumed to be nonsingular as an affine plane. In superelliptic curves, $a = 2$ implies a hyperelliptic curve, and $a = 2, b = 3$ implies an elliptic curve.

In this paper, we consider more general curves called C_{ab} curves [9]:

$$C/F_q : \sum_{0 \le i \le b, 0 \le j \le a, 0 \le ai+bj \le ab} \alpha_{i,j} X^i Y^j = 0 \ ,$$

where $\alpha_{i,j} \in F_q$, $\alpha_{b,0} \neq 0$, $\alpha_{0,a} \neq 0$, and the curve is assumed to be nonsingular as an affine plane.

Previous methods for computing Jacobians [1, 5] are based on the fact that a Jacobian group is isomorphic to the ideal class group of the coordinate ring $F_q[x, y]$ with $x = X \bmod C$ and $y = Y \bmod C$ in a canonical manner, which holds for C_{ab} curves. They reduce the problem of finding a good representative for a divisor class to that of finding a good representative of the corresponding ideal class (see Section 3).

On the other hand, Galbraith, S.Paulus, and Smart [5] reduced the problem of finding the representative element of each ideal class in a superelliptic curve to that of finding a minimal element in a lattice belonging to ideal in the ideal class, where the minimization is taken based on a certain metric suitable for superelliptic curves (see Section 4 for details), and applied an LLL-like algorithm [3] which ensures to find the minimal solution for this setting (S. Paulus [11]). In particular, in Paulus's LLL-like algorithm, division between polynomials is not required, so that Galbraith et. al's method [5] computes Jacobian group arithmetic in $O(g^2)$ operations in the base field (see Section 4 for details).

S. Arita [1] reduced the problem of finding the representative element of each ideal class for a C_{ab} curve to that of finding the minimal element in an ideal belonging to the ideal class, where the minimization is taken based on a certain monomial order suitable for C_{ab} curves (see Section 5 for details), and applied the so-called Buchberger algorithm that computes the reduced Gröbner basis. However, it is generally hard to evaluate the computational effort of finding a

Gröbner basis of an ideal in a strict manner. Even in Arita's heuristic analysis, computing Jacobian group arithmetic is supposed to take $O(g^3)$ operations in the base field.

In this paper, we generalize Galbraith et. al's method [5] to C_{ab} curves, so that there does exist a method which performs Jacobian group arithmetic on C_{ab} curves in $O(g^2)$ operations in the base field. To this end, we first point out that the lattice reduction in Galbraith et. al. [5] is essentially equivalent to the problem of finding the minimal element in an ideal with respect to the C_{ab} order. We further modify Paulus's LLL-like algorithm for the lattice using a C_{ab} curve. As a result, we prove that the modification gives a more efficient algorithm. Moreover, we propose an efficient method for computing the inverse ideal of an ideal in the coordinate ring of a C_{ab} curve (see Section 6 for details). We will see that the method proposed in [5] for computing an inverse ideal is specific to the case of superelliptic curves. On the other hand, it turns out that a certain method for computing an inverse ideal in number fields works quite well for function fields defined using a C_{ab} curve.

2 Superelliptic and C_{ab} Curves

The notation follows [13] [15].

2.1 Superelliptic Curves

Definition 1 ([5]) *A superelliptic curve defined over K is a nonsingular curve given as follows:*

$$Y^a = \sum_{0 \le i \le b} \alpha_i X^i \ , \tag{1}$$

where $\alpha_i \in K$, $\alpha_{b,0} \neq 0$, a and b are coprime, and char(K) does not divide a.

By definition, in elliptic and hyperelliptic curves we have $a = 2$, $b = 3$, and $a = 2$, $b \geq 3$, respectively. Then, the genus of a superelliptic curve is given [5] by

$$g = (a-1)(b-1)/2 \ . \tag{2}$$

2.2 C_{ab} Curves

Let C be a curve defined over K with at least one K-rational point P. Then, if we define $M_P := \{-v_P(f) | f \in L(\infty P)\}$, M_P makes a unitary semigroup under addition.

Definition 2 (C_{ab} curve) *If the semi-group M_P is generated by two positive integers a and b with* g.c.d$(a,b) = 1$, *the pair (C, P) is said a C_{ab} curve.*

Let (C, P) be a C_{ab} curve. By definition, there exist functions $X \in L(\infty P)$ and $Y \in L(\infty P)$ with pole orders a and b at P, respectively. Using these two functions X and Y, we obtain the affine model of the C_{ab} curve as follows [9]:

$$C/K : \sum_{0 \le i \le b, 0 \le j \le a, ai+bj \le ab} \alpha_{i,j} X^i Y^j = 0 \ , \tag{3}$$

where $\alpha_{i,j} \in K$, $\alpha_{b,0} \neq 0$, and $\alpha_{0,a} \neq 0$. The affine model in (3) is said the Miura canonical form of the C_{ab} curve (C, P). In the Miura canonical form, a C_{ab} curve is assumed to be nonsingular in the affine plane, and P is the only infinite place P_∞ of curve C [9].

We assume that a C_{ab} curve is given in a Miura canonical form. Then, it turns out that C_{ab} curves include superelliptic curves with the same $(a,\ b)$. In fact, superelliptic curves are C_{ab} curves with $\alpha_{i,j} = 0$ ($0 \le i \le b$ and $1 \le j \le a-1$) and $\alpha_{i,a} = 0$ ($1 \le i \le b$).

As for superelliptic curves, the formula

$$g = (a-1)(b-1)/2 \tag{4}$$

holds also for C_{ab} curves.

Definition 3 (C_{ab} order [9]) *We order as $\alpha >_{ab} \beta$ for $\alpha = (\alpha_1, \alpha_2), \beta = (\beta_1, \beta_2) \in Z^2_{\ge 0}$ if one of the following two conditions holds:*

1. *$a\alpha_1 + b\alpha_2 > a\beta_1 + b\beta_2$, or*
2. *$a\alpha_1 + b\alpha_2 = a\beta_1 + b\beta_2$, $\alpha_1 \le \beta_1$.*

By definition, under the condition $C(x, y) = 0$, monomials $X^{\alpha_1} Y^{\alpha_2}$ are ordered based on the pole order at infinity P_∞:

$$-v_{P_\infty}(X^{\alpha_1} Y^{\alpha_2}) = a\alpha_1 + b\alpha_2 \ ,$$

and if they are equal, we suppose that the larger the degree with respect to X, the smaller the monomial order.

Similarly, polynomials $f = \sum \alpha_{i,j} X^i Y^j$ can be ordered according to the pole order at infinity P_∞:

$$-v_{P_\infty}(f) = \max_{i,j,\alpha_{i,j} \neq 0} \{ai + bj\}.$$

3 Isomorphism between Jacobian and Ideal Class Groups

Jacobian group arithmetic on C_{ab} can be realized using the fact that the Jacobian group is isomorphic to the ideal class group of the coordinate ring for superelliptic and C_{ab} curves [1, 5].

Definition 4 *If $D \in Div^0_K(C)$ is expressed as $E - nP_\infty$ with $E \ge 0$ and $P_\infty \notin$ support(E), D is said a semi-reduced divisor.*

Lemma 1 ([1, 5]) *For each $j \in J_K(C)$, there exists a semi-reduced divisor $D \in Div_K^0(C)$ such that $j = [D]$.*

Definition 5 *If n is minimized in $D_1 = E - nP_\infty$ with $E \geq 0$ and $P_\infty \notin \mathrm{support}(E)$ (semi-reduced) and $D_1 \sim D \in Div_K^0(C)$, then D_1 is said the reduced divisor equivalent to D.*

Lemma 2 ([1, 5]) *If $D = E - nP_\infty \in Div_K^0(C)$ with $E \geq 0$ and $P_\infty \notin \mathrm{support}(E)$ is a reduced divisor, then the reduced divisor $D_1 \sim D$ is unique for each $D \in Div_K^0(C)$, and $\deg(E) \leq g$*

We can obtain reduced divisors using the following algorithm [1, 5]

```
Algorithm 1
Input: Semi-reduced divisor $D = E - nP_\infty \in Div_K^0(C)$ with $E \geq 0$ and $P_\infty \notin$
    support$(E)$.
Output: The reduced divisor $G \sim -D$.

Step 1: Find $f \in L(\infty P_\infty)$ satisfying $(f)_0 \geq E$ and the pole order $-v_{P_\infty}(f)$ is
    minimal, where $L(\infty P_\infty) := \cup_{i=0}^{i=\infty} L(iP_\infty)$.
Step 2: $G \leftarrow -D + (f)$.
```

Since Algorithm 1 outputs a divisor equivalent to (-1) times the input divisor, if Algorithm 1 is applied twice, a divisor equivalent to the input divisor can be obtained.

However, directly dealing with divisors is not generally efficient because of irreducible decomposition of polynomials. So, Arita [1] and Galbraith et. al. [5] independently proposed Jacobian group arithmetic using ideal representation.

Since C_{ab} curve (C, P_∞) is nonsingular in the affine plane, the coordinate ring $K[x, y]$ with $C(x, y) = 0$ is a Dedekind domain. For a C_{ab} curve (C, P_∞), an isomorphism Φ between the Jacobian group $J_K(C)$ and the ideal class group $H(K[x, y])$ of $K[x, y]$ is given as follows:

$$\Phi : J_K(C) \to H(K[x, y]) \ ,$$

$$[\sum_{P \in C, P \neq P_\infty} n_P P - (\sum_{P \in C, P \neq P_\infty} n_P) P_\infty] \mapsto [L(\infty P_\infty - \sum_{P \in C, P \neq P_\infty} n_P P)], \quad (5)$$

where we denote the ideal class which ideal $I \subset K[x, y]$ belongs to by $[I]$.

We call the ideals corresponding to reduced and semi-reduced divisors the reduced and semi-reduced ideals, respectively; then each semi-reduced ideal I is expressed by an integral ideal $I = L(\infty P_\infty - E) \subset L(\infty P_\infty) = K[x, y]$ with $E \geq 0$ and $P_\infty \notin \mathrm{support}(E)$.

Now each integral ideal of $K[x, y]$ is a $K[x]$-module, and if a $K[x]$-basis is given as $(\beta_0, \cdots, \beta_{a-1})$ with $\beta_i = \sum_{j=0}^{a-1} \beta_{i,j}(x) y^j$, the $K[x]$-basis can be uniquely expressed by taking the Hermite normal form (HNF) of the matrix $(\beta_{i,j})$ (see Appendix). Therefore, we express each representative element of an ideal class group in $K[x, y]$ by the HNF of the $K[x]$-basis.

Definition 6 *We define the degree of a (fractional) ideal in $K[x, y]$ to be a degree of x in the product of the diagonal elements (subtracted by the degree of the denominator) of the HNF.*

Then, it turns out that the degree of an ideal coincides with a value of n in the corresponding semi-reduced divisor $E - nP_\infty$. Hence, the sum of the degrees with respect to x in each column of the HNF of a reduced ideal is at most g (see Lemma 2). It is known that the product of diagonal elements in the HNF expression of I is the norm of I [5].

Hence, Algorithm 1 can be replaced by

Algorithm 2
Input: *Semi-reduced ideal I.*
Output: *The reduced ideal $J \sim I^{-1}$.*

Step 1: *Find $f \in I$, $f \neq 0$ such that the pole order $-v_{P_\infty}(f)$ is minimal.*
Step 2: $J \leftarrow (f)I^{-1}$.

4 Jacobian Group Arithmetic on Superelliptic Curves

Galbraith et. al. [5] proposed an algorithm (Algorithm 3) for performing Jacobian group arithmetic on superelliptic curves. Algorithm 3 below computes a $K[x]$-basis to represent an ideal in an ideal class: we embed $K[x, y]$ into $(K[x])^a$ with

$$\phi : K[x, y] \to (K[x])^a$$
$$\sum_{0 \leq i \leq a-1} c_i(x)y^i \mapsto (c_0(x), \cdots, c_{a-1}(x))$$

and define the metric of $C = (c_0(x), \cdots, c_{a-1}(x)) \in (K[x])^a$ as follows: $|C| := \max|C|_i$ where $|C|_i := \deg_x(c_i(x)) + \frac{b}{a}i$. Consider an ideal $I \subset K[x, y]$ and let $\{f_0, \cdots, f_{a-1}\}$ be a $K[x]$-basis of I; then, $\phi(I)$ is a lattice generated by $\{\phi(f_i)\}_i$ over $K[x]$, so that minimization over $f \in I$ with respect to $-v_{P_\infty}(f)$ is equivalent to minimization over $u \in \phi(I)$ with respect to $|u|$ ($-v_{P_\infty}(f) = a|\phi(f)|$ for $f \in I$).

Galbraith et. al. [5] apply Paulus's method [11] in the following way.

Definition 7 ([5]) *The orthogonality defect $OD(f_0, \cdots, f_{a-1})$ of a basis $\{f_0, \cdots, f_{a-1}\}$ for a lattice L is defined as*

$$OD(f_0, \cdots, f_{a-1}) := \sum_i |f_i| - \deg_x(d(L)),$$

where $d(L) := \det(f_0^*, \cdots, f_{a-1}^*)$ with $f_i^* := (f_0^i(x),\ f_1^i(x)x^{\frac{b}{a}} \cdots, f_{a-1}^i x^{\frac{b}{a}(a-1)})^t$ for $f_i = \sum_{j=0}^{a-1} f_j^i(x)y^j$.

It is easy to see that $OD(f_0, \cdots, f_{a-1}) \geq 0$.

Definition 8 ([11]) *The basis $\{f_0, \cdots, f_{a-1}\}$ for a lattice is said a reduced basis if $OD(f_0, \cdots, f_{a-1}) = 0$.*

Proposition 1 ([11]) *Let $\{f_0, \cdots, f_{a-1}\}$ be the reduced basis for an lattice L. Then $f \in \{f_0, \cdots, f_{a-1}\}$ such that $|f| = \min_i\{|f_i|\}$ is the minimal nonzero element in L with respect to $|\cdot|$.*

Algorithm 3 (Jacobian group arithmetic on superelliptic curves [5])

```
Input: Reduced ideals $I_1$, $I_2$ in $K[x, y]$ (HNF).
Output: The reduced ideal $I_3 \sim I_1 I_2$ (HNF).

Step 1: $D \leftarrow I_1 I_2$;
Step 2: $J \leftarrow$ a semi-reduced ideal equivalent to $D^{-1}$;
Step 3: $f \leftarrow$ a minimal nonzero element in $J$ with respect to $|\phi(\cdot)|$.
Step 4: $I_3 \leftarrow$ the HNF of $(f)J^{-1}$.
```

The validity of Algorithm 3 can be easily checked: basically, the process of Algorithm 2 is done twice in Steps 2-4. (Note that J in Step 2 is not required to be a reduced ideal but I_3 in Step 4 is.) We now discuss some of theses steps in detail; this will show that Algorithm 3 really uses superelliptic curves.

In Step 2, for $D = I_1 I_2$ an integral ideal equivalent to D^{-1} is computed using the formula

$$D^{-1} \sim \Pi_{\sigma \in Gal(K(x,y)/K(x)), \sigma \neq 1} D^{\sigma} \ .$$

Note that here it is assumed that K contains the a-th roots of unity. So if necessary, the base field is extended in this step. Any $\sigma \in Gal(K(x, y)/K(x))$ is given by $y^{\sigma} = \rho y$ for some a-th root of unity ρ. Hence the conjugates D^{σ} and therefore also D^{-1} are easy to compute. It seems unclear how to extend this idea to more general C_{ab} curves.

For Step 3, we can obtain the minimal element by finding the reduced basis. The complexity of finding a reduced basis is given as follows:

Proposition 2 ([11]) *We can find the reduced basis from a $K[x]$-basis $\{C_0, \cdots, C_{a-1}\}$ of the lattice in*

$$O(a^3 \max|C_i| \times OD(C_0, \cdots, C_{a-1}) \log^2 q). \tag{6}$$

For Step 4, since

$$I_3 = J^{-1}(f) = \frac{D}{\prod D^{\sigma}}(f) = \frac{I_1 I_2}{N_{K(x,y)/K(x)}(I_1 I_2)}(f) \ ,$$

and since the norm $N_{K(x,y)/K(x)}(I_1 I_2)$ is obtained computing the product of the diagonal elements in the HNF of the ideal $I_1 I_2$, the ideal I_3 can be easily computed [5].

In summary, the whole computation can be evaluated as in Proposition 3.

Proposition 3 ([5]) *Let C/K be a superelliptic curve. Jacobian group arithmetic on $Jac_K(C)$ (Algorithm 3) can be performed in $O(a^7 g^2 \log^2 q)$ if $a|q-1$ and in $O(a^9 g^2 \log^2 q)$ if $a \not| q-1$*

5 Jacobian Group Arithmetic on C_{ab} Curves

Arita[1] proposed an algorithm (Algorithm 4) for performing Jacobian group arithmetic on C_{ab} curves. Algorithm 4 below computes a $K[x, y]$-basis to represent a unique ideal in an ideal class. The idea is that in C_{ab} order, monomials are arranged according to the pole orders at infinity P_∞ when they are regarded as functions on a C_{ab} curve.

Algorithm 4 (Jacobian group arithmetic on C_{ab} curves [1])
Input: *Reduced ideals I_1, I_2 in $K[x, y]$.*
Output: *The reduced ideal I_3 equivalent to ideal product $I_1 I_2$.*

```
Step 1: $J \leftarrow I_1 I_2$;
Step 2: $f \leftarrow$ the minimal nonzero element in $J$ with respect to $C_{ab}$ order;
Step 3: $h \leftarrow$ the minimal nonzero element with respect to $C_{ab}$ order satisfying
    $(h)J \subseteq (f)$;
Step 4: $I_3 \leftarrow (h/f)J$.
```

The validity of Algorithm 4 can be easily checked: basically, the process of Algorithm 2 is done in Steps 2-4. (In particular, h and $(f)J^{-1}$ play the roles of the f and I in the second round of Algorithm 2, respectively.)

In Algorithm 4, the minimal element in an ideal is computed by finding the reduced Gröbner basis. (Note that a reduced Gröbner basis gives the unique representation of an ideal.) However, it takes much time to obtain a Gröbner basis, and it is hard to evaluate its computational effort in a strict manner. In [1], the computation of Step 2 is heuristically analyzed to be $O(g^3 \log^2 q)$ if the value of a is bounded.

However, to authors' knowledge, it seems that there has been none thus far to address Jacobian group arithmetic on C_{ab} curves except Algorithm 4 [1].

6 Fast Jacobian Group Arithmetic on C_{ab} curves

From the considerations in the previous sections, it turns out that the following two problems should be solved for extending Galbraith et. al.'s method to C_{ab} curves:

1. how to compute the inverse ideal I^{-1} given an ideal I; and
2. how to compute the minimal element over an ideal with respect to C_{ab} order.

6.1 Computing Inverse Ideals

For the first problem, we propose a more general method to obtain an inverse ideal than that in the case of superelliptic curves. The idea is based on the method for computing inverse ideals in the integral closure of a number field [3]. Let L be a number field, and Z_L the integral closure of L, and $n := [L : Q]$. We first fix the Z-basis $(w_i)_{1 \le i \le n}$ of Z_L.

Definition 9 *The different of L is defined as*

$$\varGamma(L) := \{x \in L | \mathrm{Trace}_{L/Q}(xZ_L) \subset Z\}^{-1} \ . \tag{7}$$

Then, the following proposition follows [3]:

Proposition 4 *Let $(\omega_i)_{1\le i\le n}$ be a Z-basis of Z_L and I an ideal of Z_L given by a matrix M whose columns give the coordinates of a Z-basis $(\gamma_i)_{1\le i\le n}$ of I on the chosen Z-basis. Let $T = (t_{i,j})$ be the $n \times n$ matrix such that $t_{i,j} = \mathrm{Trace}_{L/Q}(\omega_i\omega_j)$. Then, the columns of the matrix $(M^tT)^{-1}$ form a Z-basis of the ideal $(I\varGamma(L))^{-1}$.*

Therefore, for a given ideal $I \subset Z_L$, the ideal product $I\varGamma(L)^{-1}$ is computed by taking the HNF of the $n\times n^2$ matrix obtained from M and T^{-1}. If the HNF is N, then, by Proposition 4, $(N^tT)^{-1}$ forms a Z-basis of $(I\varGamma(L)^{-1})^{-1}\varGamma(L)^{-1} = I^{-1}$.

Now we go back to the case of C_{ab} curves. The ring $L(\infty P_\infty)$ is a Dedekind domain. Furthermore, since C_{ab} curves are generally nonsingular, $L(\infty P_\infty)$ coincides with the coordinate ring $K[x,y]$. Therefore, the integral closure of $K[x]$ in $K(x,y)$ is $K[x,y]$, so that the result for Z_L can be extended to $K[x,y]$ in a natural manner. Then, $1, y, \cdots, y^{a-1}$ can be the $K[x]$-basis of $K[x,y]$, and $T = (t_{i,j})_{1\le i\le a,\ 1\le j\le a}$ are given by $t_{i,j} = \mathrm{Trace}_{K(x,y)/K(x)}(y^{i+j-2})$. The value of each $t_{i,j}$ can be computed using the Newton formula (page 163, [3]) if the definition equation is given. Let $D_i(x)$ and $C_l^{(i)}$ as $y^a = \sum_{i=0}^{a-1} D_i(x)y^i$ (the definition equation of a C_{ab} curve) and $y^i = \sum_{l=0}^{a-1} C_l^{(i)}(x)y^l$ ($a \le i \le 2a-2$), respectively, in $K[x,y]$ with $x = X \bmod C$ and $y = Y \bmod C$. Then, $\mathrm{Trace}_{K(x,y)/K(x)}(1) = a$, $\mathrm{Trace}_{K(x,y)/K(x)}(y) = D_{a-1}(x)$, for $i = 2, \cdots, a-1$

$$\mathrm{Trace}_{K(x,y)/K(x)}(y^i) = iD_{a-i}(x) + \sum_{l=1}^{i-1} D_{a-l}(x)\mathrm{Trace}_{K(x,y)/K(x)}(y^{i-l}) \ , \tag{8}$$

and for $i = a, \cdots, 2a-2$

$$\mathrm{Trace}_{K(x,y)/K(x)}(y^i) = \sum_{l=0}^{a-1} C_l^{(i)}(x)\mathrm{Trace}_{K(x,y)/K(x)}(y^l) \ . \tag{9}$$

If we compute and store the matrix dT^{-1} with d the determinant of T beforehand, we obtain:

Algorithm 5 (Computation of inverse ideals for C_{ab} curves)
Input: *Semi-reduced ideal I in $K[x,y]$ with $(\gamma_i)_{1\le i\le a}$ a $K[x]$-basis of I (HNF).*
Output: *The inverse ideal I^{-1}.*

```
Step 1: $N \leftarrow$ the HNF of the $a \times a^2$ matrix $(\gamma_i\delta_j)$, with $\delta_j$ column vectors of
    $dT^{-1}$;
Step 2: $h \leftarrow \det(N^t)$;
    $P \leftarrow dh(N^tT)^{-1} = (dT^{-1})(h(N^t)^{-1})$;
    $k \leftarrow \mathrm{GCM}(\mathrm{GCM}(P),\ h)$;
```

```
$e \leftarrow \frac{h}{k}$;
$W \leftarrow \frac{1}{k}P$;
$I^{-1} \leftarrow (W, e)$ ($I^{-1} = W(e)^{-1}$).
```

(GCM(A) with A a matrix and GCM(f, g) with $f,\ g \in K[x]$ denote the GCM of all the elements in A and that of f and g, respectively.)

Theorem 1 *Algorithm 5 is computed in* $O(a^8 g^2 \log^2 q)$ *and in* $O(a^4 g^2 \log^2 q)$ *for* C_{ab} *and superelliptic curves, respectively, if the degree of an ideal* I *is* $O(g)$.

Theorem 1 is obtained based on the following facts: if the degree of x in the determinant of an $m \times n$ matrix M is bounded by t,

1. the Hermite normal form (HNF) of M with rank(M) $= m$ is obtained in $O(m^2 n t^2 \log^2 q)$ (for the proof, see Appendix);
2. if $n = m$, the determinant of M is obtained in $O(\max\{m^3 t \log^2 q,\ t^2 \log^2 q\})$ (for the proof, see Appendix);
3. if $n = m$, the inverse matrix of M is obtained in $O(\max\{m^5 t \log^2 q,\ m^2 t^2 \log^2 q\})$. (computing the m^2 determinants yields the inverse ideal if Cramer's formula is applied);

and if the degrees of x in two polynomials f, g is bounded by s,

4. the GCM of f and g is obtained in $O(s^2 \log^2 q)$.

Proof of Theorem 1:

1) General case

For Step 1, the degree of x in $\mathrm{Trace}_{K(x,y)/K(x)}(y^i)$, $0 \le i \le a-1$, is $O(g)$: in fact, from $deg_x[D_{a-l}(x)] \le b$, $0 \le l \le a-1$, and (8), we have

$$\begin{aligned} deg_x[\mathrm{Trace}_{K(x,y)/K(x)}(y^i)] &\le \max_{1 \le l \le i}\{deg_x[D_{a-l}(x)] + deg_x[\mathrm{Trace}_{K(x,y)/K(x)}(y^{i-l})]\} \\ &\le \max_{1 \le l \le i}\{b + deg_x[\mathrm{Trace}_{K(x,y)/K(x)}(y^{i-l})]\} \\ &\le ib + deg_x[\mathrm{Trace}_{K(x,y)/K(x)}(y^0)] \\ &= ib\ . \end{aligned}$$

For $a \le i \le 2a-2$, one checks that the degree of x in $C_l^{(i)}(x)$ is at most $b(i-a+1)$ (In fact, $deg_x(y^a) = b$, and $deg_x(y^i) \le b(i-a+1)$ implies $deg_x(y^{i+1}) \le b(i-a+1)+b$ for $i = a+1, \cdots, 2a-2$.), so that

$$\begin{aligned} deg_x[\mathrm{Trace}_{K(x,y)/K(x)}(y^i)] &\le \max_{0 \le l \le a-1}\{deg_x[C_l^{(i)}(x)] + deg_x[\mathrm{Trace}_{K(x,y)/K(x)}(y^l)]\} \\ &\le \max_{0 \le l \le a-1}\{b(i-a+1) + lb\} \\ &\le ib\ , \end{aligned}$$

where (9) has been applied.

In any case, the degree of x in each element of T is $ib = O(g)$ (see (4)). If we apply Cramer's formula, the degree of x of each element in dT^{-1} with $d = \det(T)$ is bounded by $O(ag)$ since the degree of x of each element in T is at most g, so is $\deg_x(\delta_j)$.

On the other hand, by assumption the degree of x in each element in the HNF expressing the input ideal is at most g, i.e, $\deg_x(\gamma_j) \leq g$. Since there are a^2 pairs of $(\gamma_i\delta_j)_{i,j}$, they are obtained in $O(a^4g^2\log^2 q)$. Using **1** with $m = a$, $n = a^2$, and $t = O(a^2g)$, the HNF N of the $a \times a^2$ matrix is obtained in $O(a^8g^2\log^2 q)$.

For Step 2, if we apply Cramer's formula, $(N^t)^{-1}$ and $\det(h)$ are computed in $O(a^7g^2\log^2 q)$ (use **3** and **2** with $m = a$ and $t = O(a^2g)$, respectively). Since the degrees of x of each element in matrices dT^{-1} and $h(N^t)^{-1}$ are $O(ag)$ and $O(a^2g)$ (note that the degree of x in each element of an HNF is at most a times as that of the original matrix), the degree of x in each element of matrix P is $O(a^2g)$. Since the GCM of two polynomials of degree $O(a^2g)$ is computed in $O(a^4g^2\log^2 q)$ (use **4** with $s = O(a^2g)$), $GCM(GCM(P), h)$ is computed in $O(a^6g^2\log^2 q)$.

Since a^2 divisions between polynomials of degree $O(a^2g)$ are done (recall that the degree of x in each element of P is $O(a^2g)$, so is the degree of x in k), W is obtained in $O(a^6g^2\log^2 q)$.

Hence, Step 2 takes $O(a^6g^2\log^2 q)$.

Therefore, Algorithm 5 takes $O(a^8g^2\log^2 q)$.

2) the case of superelliptic curves

Let $y^a = f(x)$ be a C_{ab} curve with $\deg_x f(x) = b$.

For Step 1, the HNF representation of the ideal dT^{-1} is $[f(x),\ y, \cdots,\ y^{a-1}]$. In fact, one checks

$$T = \begin{bmatrix} a & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & af(x) \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & af(x) & \cdots & 0 \\ 0 & af(x) & 0 & \cdots & 0 \end{bmatrix},\ dT^{-1} = \begin{bmatrix} f(x) & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 1 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 1 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \end{bmatrix}$$

and $d = a^a(f(x))^{a-1}$. Thus, the degree of the ideal expressed by dT^{-1} is $deg_x f(x) = b$ since the HNF of dT^{-1} is

$$\begin{bmatrix} f(x) & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & & \\ 0 & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

and the total degree of the diagonal elements is $deg_x f(x) = b$. On the other hand, by assumption, the degree of the ideal expressed by (γ_i) is $O(g)$. So, the degree of the ideal expressed by the HNF N is $O(g) + b = O(g)$. Since $(\gamma_i\delta_j)_{i,j}$ are obtained in $O(a^2g^2\log^2 q)$, from **1** with $m = a$, $n = a^2$, and $t = O(g)$, the HNF N is obtained in $O(a^4g^2\log^2 q)$. And, the degree of the ideal N is $O(g+b) = O(g)$.

For Step 2, if we apply Cramer's formula, $(N^t)^{-1}$ and $\det(h)$ are computed in $O(\max\{a^5g\log^2 q,\ a^2g^2\log^2 q\}) = O(a^4g^2\log^2 q)$ (use **3** and **2** with $m = a$ and $t = O(g)$, respectively, and note $a = O(g)$). Since the degrees of x of each element in matrices dT^{-1} and $h(N^t)^{-1}$ are $O(g)$, the degree of x in each element of matrix P is $O(g)$. Since the GCM of two polynomials of degree $O(g)$ is computed in $O(g^2\log^2 q)$ (use **4** with $s = O(g)$), $GCM(GCM(P), h)$ is computed in $O(a^2g^2\log^2 q)$.

Since a^2 divisions between polynomials of degree $O(g)$ are done (recall that the degree of x in each element of P is $O(g)$, so is the degree of x in k), W is obtained in $O(a^4g^2\log^2 q)$.

Hence, Step 2 takes $O(a^4g^2\log^2 q)$.

Therefore, Algorithm 5 takes $O(a^4g^2\log^2 q)$. □

Note that in the proof of Theorem 1, degree of x in each element of W is bounded by $O(a^2g)$ and $O(g)$ for C_{ab} and superelliptic curves, respectively, which will be referred later.

6.2 Computing the minimal element

For the second problem, by the definition of the metric $|\cdot|$ in $(K[x])^a$, for $f = \sum_{i=0}^{a-1} f_i(x)y^i \in K[x,y]$ with $f_i(x) \in K[x]$, we have

$$-v_{P_\infty}(f) = max_i\{a\deg_x(f_i(x)) + bi\} = a|\phi(f)|\ .$$

Therefore, for an ideal $I \subset K[x,y]$, minimization over I with respect to C_{ab} order is equivalent to minimization over $\phi(I)$ with respect to $|\cdot|$, so that for the second problem we can apply Paulus's method [11](finding the reduced basis) to C_{ab} curves.

Proposition 5 ([11]) *Let $b_1, \cdots, b_n$ be a basis for a lattice L and denote by $b_{i,j}$ the j-th coordinate of b_i. If the coordinates of the vectors $b_1, \cdots, b_n$ can be permuted in such a way that they satisfy*

1. *$|b_i| \le |b_j|$ for $1 \le i < j \le n$; and*
2. *$|b_{i,j}| < |b_{i,i}| \ge |b_{i,k}|$ for $1 \le j < i < k \le n$.*

Then $b_1, \cdots, b_n$ forms a reduced basis.

Now we go back to the case of C_{ab} curves. For a $K[x]$-basis $f_0, \cdots, f_{a-1}$ for a lattice L, if it satisfies that $|f_i| - |f_j| \notin Z$ $(0 \le i < j \le a-1)$, then $f_0, \cdots, f_{a-1}$ forms a reduced basis by Proposition 5. (Note that $g.c.d(a,b) = 1$ implies there exists an unique l such that $|f| = |f|_l$ for a nonzero vector $f = (f_0, \cdots, f_{a-1}) \in (K[x])^a$. In fact, if $|f| = |f|_i = |f|_j$ with $0 \le i \le j \le a-1$, i.e. $ac_i + bi = ac_j + bj$, where c_i and c_j are the degrees of x in $f_i(x)$ and $f_j(x)$, respectively, then $a(c_i - c_j) = b(j-i)$. Hence, $a|j-i$ since $g.c.d(a,b) = 1$, which implies $i = j$.)

Therefore, we can modify Paulus's algorithm [11] to obtain the following algorithm.

Algorithm 6 (Computation of reduced basis in C_{ab} curves)
Input: *$K[x]$-basis $\{f_0, \cdots, f_{a-1}\}$ for a lattice L with $f_i = (f_{i,0}(x), \cdots, f_{i,a-1}(x))^t$.*
Output: *The reduced basis.*

```
Step 1: $g_0 \leftarrow f_0,\ k \leftarrow 1$;
Step 2: $g_k \leftarrow f_k$;
Step 3: if $|g_j| - |g_k| \notin Z\ (\forall j < k)$ then $k \leftarrow k+1$, otherwise go to Step 5-1;
Step 4 if $k = a$ then output $\{g_0, \cdots, g_{a-1}\}$, otherwise go to Step 2;
Step 5-1 let $j,\ l$ be the indices such that $|g_j| - |g_k| \in Z$, $|g_j| = |g_j|_l$, $|g_k| = |g_k|_l$;
Step 5-2 if $|g_j| > |g_k|$ then swap $g_j$ and $g_k$;
Step 5-3 $g_k \leftarrow g_k - rx^{|g_k|-|g_j|}g_j$ with $r = c_{k,l}/c_{j,l}$, where $c_{k,l}$ and $c_{j,l}$ are the
    leading coefficients of $g_{k,l}(x)$ and $g_{j,l}(x)$, respectively; and
Step 6 if $\sum_{j=0}^{k} |g_j| + \sum_{j=k+1}^{a-1} |f_j| = \deg_x d(L)$ then output $\{g_0, \cdots, g_k,\ f_{k+1}, \cdots, f_{a-1}\}$,
    otherwise go to Step 3.
```

The validity of Algorithm 6 can be checked by Definition 8 and Proposition 5.

Theorem 2 *Algorithm 6 is computed in $O(a^3t(t+b)\log^2 q)$ if the degree of x in $(f_{i,j})_{i,j}$ is bounded by t.*

Proof of Theorem 2:
It is easy to check that Step 5-3 dominates the computational complexity of Algorithm 6. In Step 5-3, the computation of $g_k \leftarrow g_k - rx^{|g_k|-|g_j|}g_j$ requires shift operations and $O(at)$ multiplications in K. Note that $OD(g_0, \cdots, g_k,\ f_{k+1}, \cdots f_{a-1})$ strictly decreases after executing Step 5-3. Therefore, the number of iterations of executing Step 5-3 is bounded by $a \times (OD(f_0, \cdots f_{a-1}) - \deg_x d(L)) \le a \times OD(f_0, \cdots f_{a-1}) = O(a(\sum_{i=0}^{a-1}(t+b))) = O(a^2(t+b))$. Hence, Algorithm 6 is computed in $O(at\log^2 q \times O(a^2(t+b))) = O(a^3t(t+b)\log^2 q)$ $\square$

For Steps 3 and 5, in [5], Paulus's original algorithm was directly applied in a straightforward manner that a set of linear equations

$$\sum_{j=0}^{k-1} c^*_{j,i} r_j = c^*_{k,i} \quad (0 \le i \le k-1)$$

is solved for r_j, $j = 0, \cdots, k-1$, every time k and $OD(f_0, \cdots, f_{a-1})$ are updated, where $c^*_{j,i}$ is the leading coefficient of $g_{j,i}(x)$ at the order of the leading term of in g_j with respect to C_{ab} order (if no such a coefficient exists in $g_{j,i}(x)$, then $c^*_{j,i} = 0$) and $c^*_{j,i} = 0$ for $0 \le j < i \le k-1$ (if necessary, swap rows in each g_j), so that the leading term in g_k can be cancelled out with either of $g_0, \cdots, g_{k-1}$. They estimated the complexity of solving the equations as $O(k^2)$ operations in the base field since the coefficient matrix $(c^*_{j,i})_{0 \le i,j \le k-1}$ is a lower triangular matrix (thus, that of computing a reduced basis as $O(a^7g^2\log^2 q)$), which we considered too large for implementation. In this paper, we find that the solution is quite simple, i.e. we only solve one linear equation (see Step 5-3) since all r_j except one are equal to zeros, which is complited in $O(1)$ operations in the base field. In fact, we have

1. for each column in the coefficient matrix $(c_{j,i})_{0\le i,j\le k-1}$ and the column vector $(c_{k,i})_{0\le i\le k-1}$, all the elements except one are equal to zeros; and
2. for each row in the coefficient matrix $(c_{j,i})_{0\le i,j\le k-1}$, all the elements except one are equal to zeros.

(apparently, $c_{j,i} = c^*_{j,i} \Leftrightarrow c^*_{j,i} \neq 0 \Leftrightarrow |g_j| = |g_j|_i$), where the first property is from the assumption $g.c.d(a,b) = 1$, and the second from Step 3 in Algorithm 6, i.e. $|g_j| - |g_i| \notin Z$ $(0 \le i < j < k-1)$.

Algorithm 6 utilizes these property, so that the computational effort has been greatly saved.

In summary, we see that the extension of Galbraith et. al.'s method to C_{ab} curves is possible. The whole proposed algorithm can be described as in Algorithm 7.

Algorithm 7 (proposed Jacobian group arithmetic on C_{ab} curves)
Input: *Ideals I_1, I_2 in $K[x,y]$ (HNF).*
Output: *The reduced ideal I_3 equivalent to I_1I_2 (HNF).*

```
Step 1: $J \leftarrow$ the HNF of $I_1I_2$;
Step 2: Applying Algorithm 5 to $J$, $J^{-1} \leftarrow (W,e)$;
Step 3: Applying Algorithm 6 to $W$, $f \leftarrow$ the minimal element in $W$ with
    respect to $C_{ab}$ order;
Step 4: $I_3 \leftarrow$ the HNF of $(f)W^{-1} = (f/e)J$.
```

Our final task is to ensure that Algorithm 7 is completed in $O(g^2)$ operations in the base field if the sizes of a and q are bounded, which is the goal of this paper. The computation time of Steps 1 and 4 is basically the same as those in Algorithm 3, which is within $O(g^2)$ operations in the base field. Step 2 is completed in $O(a^8g^2\log^2 q)$ by Theorem 1 since the degree of ideal J is $O(g)$. For Step 3, since the degree of x in each element of the matrix W is $O(a^2g)$, Step 3 is completed in $O(a^3\cdot a^2g\cdot(a^2g+b)\cdot\log^2 q) = O(a^7g^2\log^2 q)$ by Theorem 2. Hence we obtain:

Theorem 3 *Algorithm 7 is completed in $O(a^8g^2\log^2 q)$ and in $O(a^4g^2\log^2 q)$ for C_{ab} and superelliptic curves, respectively.*
(See Table 1 for details.)

7 Concluding Remarks

We proposed a fast Jaconbian group arithmetic algorithm for C_{ab} curves (Algorithm 7), evaluated the complexity of the proposed algorithm. As a result, it

Table 1. Complexity of Jacobian Group Arithmetic

	Proposed method		Galbraith, Paulus and Smart [5]
	C_{ab}	superelliptic	superelliptic
Step 1 (ideal product)	$O(a^4g^2\log^2 q)$	$O(a^4g^2\log^2 q)$	$O(a^4g^2\log^2 q)$
Step 2 (inverse ideal)	$O(a^8g^2\log^2 q)$	$O(a^4g^2\log^2 q)$	$O(a^7g^2\log^2 q)$ ($O(a^9g^2\log^2 q)$)
Step 3 (minimal element)	$O(a^7g^2\log^2 q)$ (substitute $t = a^2g$ to Theorem 2)	$O(a^3g^2\log^2 q)$ (substitute $t = g$ to Theorem 2)	$O(a^7g^2\log^2 q)$ (apply Proposition 2)
Step 4 (ideal product)	$O(a^7g^2\log^2 q)$	$O(a^4g^2\log^2 q)$	$O(a^4g^2\log^2 q)$
whole process	$O(a^8g^2\log^2 q)$	$O(a^4g^2\log^2 q)$	$O(a^7g^2\log^2 q)$ ($O(a^9g^2\log^2 q)$)

turned out that Algorithm 7 is more efficient than Algorithms 3 (Galbraith et. al.) in the case of superelliptic curves (Proposition 3, Theorem 3). Furthermore, although Algorithm 7 can be applied to C_{ab} curves as well as superelliptic curves, Algorithm 7 completes the arithmetic in $O(g^2)$ operations in the base field while Algorithm 4 does in $O(g^3)$ operations in the base field.

Future work includes exploring a faster Jacobian group arithmetic scheme for more general curves.

Acknowledgements

The authors would like to thank Shinji Miura and Junji Shikata for their useful comments.

References

1. Seigo Arita, *Algorithms for Computations in Jacobian Group of C_{ab} Curve and Their Application to Discrete-Log Based Public Key Cryptosystems*, IEICE Trans part A Vol. J82-A No.8, 1291-1299, Aug. 1999. in Japanese.
2. D. G. Cantor, *Computing in the Jacobian of a hyper-elliptic curves*, Math.Comp, **48** (1987), pp. 95-101.
3. H. Cohen, *A Course in Computational Algebraic Number Theory*, Springer-Verlag, GTM 138, 1993.
4. G. Frey and H. Rück, *A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves*, Mathematics of Computation **62** (1994), 865-874.
5. S. D. Galbraith, S. Paulus, and N. P. Smart, *Arithmetic on Superelliptic Curves*, preprint, 1998.

6. R. Hartshorne, *Algebraic Geometry*, Springer-Verlag, GTM 52, 1977.
7. N. Koblitz, *Hyperelliptic cryptosystems*, J. Cryptography, Vol. 1, 139-150, 1989.
8. V. S. Miller, *Use of elliptic curves in cryptography*, Advances in Cryptography CRYPTO '85 (Lecture Notes in Computer Science, vol 218), Springer-Verlag, 1986, pp. 417-426.
9. Shinji Miura, *The study of error coding codes based on algebraic geometry*, Dr. thesis. in Japanese (1997).
10. Achim Müller, *Effiziente Algorithmen für Probleme der linearen Algebra über Z*, Master's thesis, Universität des Saarlandes, Saarbrücken, 1994.
11. S. Paulus, *Lattice basis reduction in function field* in Ants-3, Algorithmic Number Theory(Lecture Notes in Computer Science, vol 1423), 567-575, 1998.
12. S. Paulus and A.Stein, *Comparing Real and Imaginary Arithmetics for Divisor Class Groups of Hyperelliptic Curves* in Ants-3, Algorithmic Number Theory(Lecture Notes in Computer Science, vol 1423), 576-591, 1998.
13. J. H. Silverman, *The Arithmetic of Elliptic Curves*, Graduate Texts in Math., vol. 106, Springer-Verlag, Berlin and New York, 1994.
14. N. P. Smart, *On the performance of Hyperelliptic Cryptosystems*, Advances in Cryptology EUROCRYPTO'99 (Lecture Notes in Computer Science vol 1592), 165-175, 1998.
15. H. Stichtenoth, *Algebraic Function Fields and Codes*, Springer Universitext, Springer-Verlag, 1993.

Appendix : Hermite Normal Form (HNF) with $K[x]$ Coefficients

Definition 10 *We say that an $m \times n$ matrix $A = (a_{i,j})$ with $K[x]$ coefficients is an Hermite normal form (HNF) if there exists $r \leq n$ and a strictly increasing map f from $[r+1,\ n]$ to $[1,\ m]$ satisfying the following properties:*

1. *for $r+1 \leq j \leq n$, $a_{f(j),j} \neq 0$, $a_{i,j} = 0$ if $i > f(j)$; and for $k < j$*
   (a) $\deg_x(a_{f(k),j}) < \deg_x(a_{f(k),k})$ if $\deg_x(a_{f(k),k}) \geq 1$; or
   (b) $a_{f(k),j} = 0$ if $\deg_x(a_{f(k),k}) = 0$ (equivalently, $a_{f(k),k} \in K$)
2. *the first r columns of A are equal to 0.*
3. *$a_{f(k),k}$, $k = r+1,\ \cdots,\ n$, are monic.*

Proposition 6 *Let $A = (a_{i,j})$ be an $m \times n$ matrix with $K[x]$ coefficients. Then, there exists a unique $m \times n$ matrix B in HNF of the form $B = AU$ with $U \in \mathrm{GL}_n(K[x])$, where $\mathrm{GL}_n(K[x])$ is the group of matrices with $K[x]$ coefficients which are invertible, i.e. whose determinant belongs to K.*

We call the matrix consisting of the last $n-r$ columns the HNF of A.

When we compute an HNF directly, it is hard to evaluate its complexity since we don't know how large the degree of x grows during the process. But, in the case of integer coefficients and $\mathrm{rank}(A) = m$, if we know the value D that is a multiple of the determinant of the Z-module $L(A)$ generated by the columns of A, then we can compute the HNF of A by using D [3]. And this modified method requires $O(m^2 n |D|^2)$-bit operations, where $|D|$ is the number of bits for expressing D. (Note that in the case of a finite field, the computation of an HNF takes $O(m^2 n)$ operations in the field [3].) Therefore we obtain the following algorithm by extending the result for Z to $K[x]$ in a natural manner.

```
Algorithm 8 HNF
Input: $m \times n$ matrix $A$ with $K[x]$ coefficients and $\mathrm{rank}(A) = m$.
Output: The HNF of $A$.
Step 1: the $R \leftarrow$ the $m \times m$ matrix whose columns consist of linear independent
    column vectors of $A$;
Step 2: $D \leftarrow \det(R)$;
Step 3: Compute the HNF modulus $D$ [3];
```

Remark 1 *1. In the case of* $m = n$, *Step* 1 *is not required.*
2. In Step 2, *since* $L(R)$ *is an sub-module of* $L(A)$, *the value of* D *is a multiple of* $\det(L(A))$, *where* $(L(A))$ *is a* $K[x]$*-module generated by the columns of* A.

Proposition 7 *We assume the degree of* x *in the determinant of* A *is less than* t. *If* $q > t$, *then Algorithm 8 is completed in* $O(m^2 n t^2 \log^2 q)$.

Remark 2 *We consider the case where* g *is extraordinarily large, say* $q = 2^{160}$ *(common in cryptography etc.), so that the condition* $q > t$ *is always cleared. Otherwise, no computational problem arises.*

Proof:

For Step 1, let $a_1, \cdots, a_n$ be the column vectors which A consists of, and $A_i = [a_1, \cdots, a_i]$ be the matrix that consists of the first i columns of A. We consider $W \subset K$ of cardinality t (such a W always exists because $\#(W) = t < q = \#(K)$). Then, we have

$$\deg_x(\det(L(A))) < t = \deg_x(\Pi_{\alpha \in W} f_\alpha) \ . \tag{10}$$

Let $r_\alpha(i) := \mathrm{rank}_{K[x]/(f_\alpha(x))}(A_i \bmod f_\alpha(x))$. Then, we can show that there exists an $f_\alpha(x)$ such that $\mathrm{rank}(A) = r_\alpha(n)$. Suppose $\mathrm{rank}(A) < r_\alpha(n)$ for all $\alpha \in W$. Then, $\det(L(A)) \bmod f_\alpha = 0$ for all $\alpha \in W$. But, this implies $\Pi_{\alpha \in W} f_\alpha$ divides $\det(L(A))$, which contradics (10).

So, we can construct linear independent column vectors of A, i.e. Step 1 can be broken down into the following stages:

Stage 1 choose an $f_\alpha \in W$, and for each $1 \le i \le n$ compute $r_\alpha(i)$;
Stage 2 if there exists an l such that $r_\alpha(l) = m$, go to Stage 4-1;
Stage 3 $W \leftarrow W - \{f_\alpha\}$ and go to Stage 1;
Stage 4-1 if $r_\alpha(1) = 1$, then choose a_1, otherwise throw away a_1; and
Stage 4-2 for each $2 \le i \le l$, choose a_i such that $r_\alpha(i-1) < r(i)$.

It is clear that the computation of Stage 1 dominates Step 1. We can obtain the value of $r_\alpha(i)$, $1 \le i \le n$, by computing the HNF of the $a \times a^2$ matrix $A \bmod f_\alpha(x)$. From $K[x]/(f_\alpha(x)) \cong K$ and the fact that the number of iterations in Stage 1 is bounded by $\#(W)$, it turns out that Step 1 takes

$\#(W) \times O(m^2 n \log^2 q) = O(t) \times O(m^2 n \log^2 q) = O(m^2 nt \log^2 q)$ (the HNF is obtained in $O(m^2 n \log^2 q)$ if each element of the element is in K [3], which is much smaller than that for $K[x]$).

For Step 2, we can obtain the value of D by computing $D \bmod f_\alpha(x)$ for each $\alpha \in W$ and applying the Chinese Remainde Theorem. It takes $\#(W) \times O(m^3 \log^2 q) = O(m^3 t \log^2 q)$ to compute $D \bmod f_\alpha(x)$ for all $\alpha \in W$. Then, $D = \Sigma_\alpha g_\alpha(D \bmod f_\alpha(x))$, where $g_\alpha = s_\alpha h_\alpha$ with $r_\alpha f_\alpha + s_\alpha h_\alpha = 1$, where $r_\alpha \in K[x]$ and $h_\alpha = \Pi_{\alpha' \in W} f_{\alpha'} / f_\alpha$. The multiplication $\Pi_{\alpha \in W} f_\alpha$ is done in $O(\Sigma_{i=1}^{\#(W)} i \cdot \log^2 q) = O(t^2 \log^2 q)$; the division between $\Pi_{\alpha \in W} f_\alpha$ and f_α is done in $O(t \cdot \log^2 q)$ since the degrees of x in the two polynomials are $t-1$ and 1; s_α is computed in $O(1 \cdot \log^2 q)$ (Proposition 3 [12]); the multiplication $s_\alpha h_\alpha$ is done in $O(t \log^2 q)$ since the degrees of x in s_α and h_α are 0 and $t-1$; and the final computation $\Sigma_\alpha g_\alpha(D \bmod f_\alpha(x))$ takes $\#(W) \times O(1 \cdot (t-1) \log^2 q) = t \times O(t \log^2 q)$ since the degrees of x in h_α is $t-1$ and $D \bmod f_\alpha(x) \in K$. Hence, Step 2 takes $O(\max\{m^3 t \log^2 q, t^2 \log^2 q\})$.

Since the number of bits expressing D is $O(t \log q)$, Step 3 takes $O(m^2 n (t \log q)^2) = O(m^2 n t^2 \log^2 q)$ [3].

Since $m \leq n$, Algorithm 8 is completed in $O(m^2 n t^2 \log^2 q)$. $\square$