Efficient computation of Cantor's division polynomials of hyperelliptic curves over finite fields

Elie Eid

Efficient computation of Cantor's division polynomials of hyperelliptic curves over finite fields

Elie Eid

IRMAR, Université de Rennes 1, France

ARTICLE INFO

Keywords:
p-adic differential equations
Newton scheme
Arithmetic geometry
Isogenies
Cantor's polynomials

ABSTRACT

Let p be an odd prime number. We propose an algorithm for computing rational representations of isogenies between Jacobians of hyperelliptic curves via p-adic differential equations with a sharp analysis of the loss of precision. Consequently, after having possibly lifted the problem in the p-adics, we derive fast algorithms for computing explicitly Cantor's division polynomials of hyperelliptic curves defined over finite fields.

1. Introduction

An important aspect in the study of principally polarized abelian varieties over finite fields is to design effective algorithms to calculate the number of points on these varieties. In 1985, Schoof proposed the first deterministic polynomial time algorithm for counting points on elliptic curves [1]. A few years later, improvements were made by computing kernels of isogenies, resulting in the Schoof-Elkies-Atkin algorihm which is sufficiently fast for practical purposes [2, 3, 4]. In 1990, Pila gave a generalization of the classical Schoof algorithm to abelian varieties and in particular Jacobians of curves over finite fields [5]. His algorithm remains impractical in the general case but improvements were made for varieties of small dimension, typically for Jacobians of genus 2 and 3 curves [6, 7, 8]. When the inputs are Jacobians of hyperelliptic curves, isogenies (for curves of low genus) and Cantor's division polynomials (for curves of arbitrary genus) are important ingredients to these algorithms. For this reason, a keen interest has been raised to compute them efficiently [9, 10, 11, 12, 13]. In this work, we tackle, in all generality, the problem of effective computation of isogenies between Jacobians of hyperelliptic curves to obtain fast algorithms that compute Cantor's division polynomials.

1.1. Isogenies and (p-adic) differential equations

A separable isogeny between Jacobians of hyperelliptic curves of genus g defined over a field k is characterized by its so called rational representation (see Section 3.2 for the definition); it is a compact writing of the isogeny and can be expressed by $2g$ rational fractions defined over a finite extension of k. These rational fractions are related. In fields of characteristic different from 2, they can be determined by computing an approximation of the solution $X(t) \in k'[\![t]\!]^g$, where k' is a finite extension of k of degree at most $O(g!)$, of a first order non-linear system of differential equations of the form

$$H\left(X(t)\right) \cdot X'(t) = G(t) \tag{1}$$

ORCID(s):

where $H : k'[\![t]\!]^g \to M_g\big(k'[\![t]\!]\big)$ is a well chosen map and $G(t) \in k'[\![t]\!]^g$. This approach is a generalization of the elliptic curves case [10] for which Equation (1) is solved in dimension one.

Equation (1) was first introduced in [11] for genus two curves defined over finite fields of odd characteristic and solved in [14] using a well-designed algorithm based on a Newton iteration; this allowed them to compute $X(t)$ modulo $t^{O(\ell)}$ in the case of an (ℓ,ℓ)-isogeny for a cost of $\tilde{O}(\ell)$ operations in k then recover the rational fractions that defines the rational representation of the isogeny. This approach does not work when the characteristic of k is positive and small compared to ℓ, in which case divisions by p occur and an error can be raised while doing the computations. We take on this issue similarly as in the elliptic curve case [15, 12] by lifting the problem to the p-adics. We will always suppose that the lifted Jacobians are also Jacobians for some hyperelliptic curves. It is relevant to assume this, even though it is not the generic case when g is greater than 3[1] [16], since it allows us to compute efficiently the rational representation of the multiplication by an integer in which case the lifting can be done arbitrarily. After this process, we need to analyze the loss of p-adic precision in order to solve Equation (1) without having a numerical instability. We extend the result of [10] to compute isogenies between Jacobians of hyperelliptic curves, by proving that the number of lost digits when computing an approximation of the solution of Equation (1) modulo $t^{O(g\ell)}$, stays within $O(\log_p(g\ell))$ (see Sections 2 and 4).

1.2. Computing with p-padic numbers

We introduce the computation model that we will use throughout this paper. Let p be a prime number and K a finite extension of the p-adic field $\mathbb{Q}_p$. We denote by v_p the unique normalized extension to K of the p-adic valuation. We denote by $\mathcal{O}_K$ the ring of integers of K, $\pi \in \mathcal{O}_K$ a fixed uniformizer of K and e the ramification index of the extension $K/\mathbb{Q}_p$. We naturally extend the valuation v_p to quotients of $\mathcal{O}_K$, the resultant valuation is also denoted by v_p.

From an algorithmic point of view, p-adic numbers behave like real numbers: they are defined as infinite sequences of digits that cannot be handled by computers. It is thus necessary to work with truncations. For this reason, several computational models were suggested to tackle these issues (see [17] for more details). In this paper, we use the *fixed point arithmetic model* at precision $O(p^M)$, where $M \in \mathbb{N}^*$, to do computations in K. More precisely, an element in K is represented by an interval of the form $a + O(p^M)$ with $a \in \mathcal{O}_K/\pi^{eM}\mathcal{O}_K$. We define basic arithmetic operations on intervals in an elementary way

$$\begin{aligned}\big(x + O(p^M)\big) \pm \big(y + O(p^M)\big) &= (x \pm y) + O(p^M),\\ \big(x + O(p^M)\big) \times \big(y + O(p^M)\big) &= xy + O(p^M).\end{aligned}$$

For divisions we make the following assumption: for $x, y \in \mathcal{O}_K/\pi^{eM}\mathcal{O}_K$, the division of $x+O(p^M)$ by $y+O(p^M)$ raises an error if $v_p(y) > v_p(x)$, returns $0+O(p^M)$ if $x = 0$ in $\mathcal{O}_K/\pi^{eM}\mathcal{O}_K$ and returns any representative $z + O(p^M)$ with the property $x = yz$ in $\mathcal{O}_K/\pi^{eM}\mathcal{O}_K$ otherwise.

Matrix computation

We extend the notion of intervals to the K-vector space $M_{n,m}(K)$: an element in $M_{n,m}(K)$ of the form $A + O(p^M)$ represents a matrix $\big(a_{ij} + O(p^M)\big)_{ij}$ with $A = (a_{ij}) \in M_{n,m}\big(\mathcal{O}_K/\pi^{eM}\mathcal{O}_K\big)$.

[1] Indeed, the dimension of the moduli scheme $\mathcal{M}_g$ is equal to $3g-3$, while the subspace of hyperelliptic curves in it has dimension $2g-1$.

Operations in $M_{n,m}(K)$ are defined from those in K:

$$\begin{aligned}\left(A + O(p^M)\right) \pm \left(B + O(p^M)\right) &= (A \pm B) + O(p^M),\\ \left(A + O(p^M)\right) \cdot \left(B + O(p^M)\right) &= (A \cdot B) + O(p^M).\end{aligned}$$

For inversions, we use standard Gaussian elimination.

Proposition . *[18, Proposition 1.2.4 and Théorème 1.2.6] Let $A \in GL_n(\mathcal{O}_K)$ with entries known up to precision $O(p^M)$. The Gauss-Jordan algorithm computes the inverse A^{-1} of A with entries known with the same precision as those of A using $O(n^3)$ operations in K.*

1.3. Main result

We are interested in designing fast algorithms that solve Equation (1). In a first step, we arrive at a generic algorithm for solving the differential system. Its complexity depends on the complexity of matrix multiplication and the composition $H(X(t))$. Let $\mathrm{MM}(g, n)$ be the number of arithmetical operations required to compute the product of two $g \times g$ matrices containing polynomials of degree bounded by n. Our first theorem is the following.

Theorem A (See Theorem 3 and Proposition 4)**.** *Let p be a prime number and $g \geq 1$ be an integer. Let K be a finite extension of $\mathbb{Q}_p$ and $\mathcal{O}_K$ be its ring of integers. There exists an algorithm that takes as input:*

- *two positive integers n and N,*
- *an analytic map $H : \mathcal{O}_K[\![t]\!]^g \to M_g\left(\mathcal{O}_K[\![t]\!]\right)$ of the form $H(y_1(t), \cdots, y_g(t)) = (f_{ij}(y_i(t)))_{ij}$ with $f_{ij} \in \mathcal{O}_K[\![t]\!]$ and $H(0) \in GL_g(\mathcal{O}_K)$,*
- *a vector $G(t) \in \mathcal{O}_K[\![t]\!]^g$,*

and, assuming that the unique solution of the differential equation

$$H\left(X(t)\right) \cdot X'(t) = G(t)$$

is in $\left(t\mathcal{O}_K[\![t]\!]\right)^g$, outputs an approximation of this solution modulo (p^N, t^{n+1}) for a cost $O\left(\mathrm{MM}(g, n) + \mathrm{C}_H(n)\right)$ operations in $\mathcal{O}_K$, where $\mathrm{C}_H(n)$ denotes the algebraic complexity of an algorithm computing the composition $H(X(t)) \mod t^{n+1}$, at precision $O(p^M)$ with $M = \max(N, 3) + \lfloor \log_p(n) \rfloor$ if $p = 2$, $M = \max(N, 2) + \lfloor \log_p(n) \rfloor$ if $p = 3$ and $M = N + \lfloor \log_p(n) \rfloor$ otherwise.

It is important to know that one can do a bit better for $p = 2$ and 3 if we follow the same strategy as [10], in this case M is equal to $\max(N, 2) + \lfloor \log_p(n) \rfloor$ if $p = 2$ and $N + \lfloor \log_p(n) \rfloor$ otherwise. For the sake of simplicity, we will not prove this here.

The field K in the algorithm of Theorem A is generally given with a uniformizing parameter to avoid the calculation of its ring of integers. However, for the computation of isogenies, this ring is already known since the extension K can always be chosen to be unramified.

If the integer g is assumed to be small then the function $\mathrm{C}_H(n)$ depends only on the number of arithmetical operations required to compute the composition of two power series modulo t^{n+1}. In

the general case, Kedlaya and Umans bound [19] allows us to obtain $\mathrm{C}_H(n) = \tilde{O}(n)$. In our context, the composition $H(X(t)) \mod t^{n+1}$ is easy to compute since H only includes univariate rational fractions of radicals of constant degree, therefore $\mathrm{C}_H(n) = \mathrm{M}(n)$, where $\mathrm{M}(n)$ is the number of arithmetical operations required to compute the product of two polynomials of degrees bounded by n. Moreover, $\mathrm{MM}(g, n) = \mathrm{M}(n)$, hence if g is small, the algorithm in Theorem A requires at most $\tilde{O}(n \cdot [K : \mathbb{Q}_p])$ operations in $\mathbb{Z}_p$ to compute $X(t) \mod t^{n+1}$.

If g is arbitrary then the algorithm of Theorem A requires at most $\tilde{O}(g^\omega n \cdot [K : \mathbb{Q}_p])$ operations in $\mathbb{Z}_p$, where $\omega \in]2, 3]$ is a feasible exponent of matrix multiplication. This complexity can be reduced to $\tilde{O}(gn \cdot [K : \mathbb{Q}_p])$ operations in $\mathbb{Z}_p$ if H is given by a structured matrix.
In the case of the computation of a rational representation of an isogeny I over an unramified extension K_0 of $\mathbb{Q}_p$, the field K can be chosen to be an extension of K_0 of degree at most $O(g)$ and H is given by an alternant matrix. Therefore, the algorithm of Theorem A performs at most $\tilde{O}(g^2 n \cdot [K_0 : \mathbb{Q}_p])$ operations in $\mathbb{Z}_p$ to compute an approximation of $X(t) \mod t^{n+1}$; but this is not optimal in g since I is defined over K_0.
In order to remedy this problem, we work directly on the first Mumford coordinate of a rational representation of I, *i.e.* the degree g monic polynomial whose roots are the components of the solution $X(t)$, which has the decisive advantage to be defined over the base field K_0. Consequently, we obtain a fast algorithm for computing a rational representation of I in quasi-linear time. This is the main result of Section 4.

Theorem B (See Theorem 30 and Proposition 31)**.** *Let p be an odd prime number. Let K_0 be an unramified extension of $\mathbb{Q}_p$ and $\mathcal{O}_{K_0}$ the ring of integers of K_0. There exists an algorithm that takes as input:*

- *three positive integers g, n and N,*
- *a monic polynomial $U_0(z) = \prod_{j=1}^{g} (z - x_j^{(0)}) \in \mathcal{O}_{K_0}[z]$ such that $U_0(z) \mod p$ is separable,*
- *a polynomial $V_0 \in \mathcal{O}_{K_0}[z]$ of degree $g-1$ such that $V(x_j^{(0)}) \mod p \neq 0$ for all $j \in \{1, \dots, g\}$,*
- *a polynomial $f \in \mathcal{O}_{K_0}[\![t]\!][z]$ of degree $O(g)$ such that U_0 divides $f - V_0^2$ in $\mathcal{O}_{K_0}[\![t]\!][z]$,*
- *a vector $G(t) \in \mathcal{O}_{K_0}[\![t]\!]^g$,*

and, assuming that the unique solution $X(t)$ of the differential equation

$$\begin{cases} H(X(t)) \cdot X'(t) = G(t), \quad X(0) = (x_1^{(0)}, \cdots, x_g^{(0)}), \\ y_j(t)^2 = f(x_j(t)), \quad y_j(0) = V_0(x_j^{(0)}) \quad \text{for } j = 1, \dots, g \end{cases}$$

where $H(X(t))$ is the matrix defined by

$$H(X(t)) = \left(\frac{x_j(t)^{i-1}}{y_j(t)} \right)_{1 \le i,j \le g},$$

is in $\mathcal{O}_K[\![t]\!]^g$*, where* K *denotes the splitting field of* U_0*, outputs a polynomial* $U(t,z) = \prod_{i=1}^{g}(z - x_i(t)) \in \mathcal{O}_{K_0}[\![t]\!][z]$ *such that* $(x_1(t), \dots, x_g(t))$ *is an approximation of this solution modulo* (p^N, t^{n+1}) *for a cost* $\tilde{O}(ng)$ *operations in* $\mathcal{O}_{K_0}$*, at precision* $O(p^{M+\lfloor \log_p(2g-1)\rfloor})$*, with* $M = \max(N,2) + \lfloor \log_p(n)\rfloor$ *if* $p = 3$ *and* $M = N + \lfloor \log_p(n)\rfloor$ *otherwise.*

1.4. Computing Cantor's division polynomials

Cantor's division polynomials are defined as being the numerators and denominators of the components of a rational representation of the multiplication by an integer endomorphism. They were first introduced for elliptic curves and later were described for hyperelliptic curves by Cantor [20]. They are crucial in point counting algorithms on elliptic and hyperelliptic curves. Classical algorithms for computing a rational representation of the multiplication endomorphism are usually based on Cantor's paper [20] and Cantor's algorithm for adding points on Jacobians (see for example [21]). Although, they exhibit acceptable running time in practice, their theoretical complexity has not been well studied yet and experiments show that they become much slower when the degree or the genus get higher.

Using the algorithm of Theorem B, we derive a fast algorithm to compute Cantor's division polynomials over finite fields of odd characteristic. Our final result is then the following.

Theorem C (See Theorem 34)**.** *Let* p *an odd prime number and* $g > 1$ *an integer. Let* ℓ *be an integer greater than* g *and coprime to* p*. Let* $C : y^2 = f(x)$ *be a hyperelliptic curve of genus* g *defined over a finite field* k *of characteristic* p*. There exists an algorithm that computes Cantor* ℓ*-division polynomials of* C*, performing at most* $\tilde{O}(\ell^2 g^2)$ *operations in* k*.*

2. Solving a system of p-adic differential equations: the general case

In this section, we give a proof of Theorem A by solving the nonlinear system of differential equations (1) in an extension of $\mathbb{Q}_p$ for all prime numbers p. We use the computational model introduced in Section 1.2 in our algorithm exposed in Section 2.1 and the proof of its correctness is presented in Section 2.2.

Throughout this section the letter p refers to a fixed prime number and K corresponds to a fixed finite extension of $\mathbb{Q}_p$. We denote by $\mathcal{O}_K$ the ring of integers of K, $\pi \in \mathcal{O}_K$ a fixed uniformizer and e the ramification index of the extension $K/\mathbb{Q}_p$.

2.1. The algorithm

Let g be a positive integer, $K[\![t]\!]$ be the ring of formal series over K in t. We denote by $M_g(k)$ the ring of square matrices of size g over a field k. Let $\mathrm{f} = \left(f_{ij}\right)_{i,j} \in M_g\left(K[\![t]\!]\right)$ and H_{f} be the map defined by

$$\begin{array}{rcl} \left(tK[\![t]\!]\right)^g & \xrightarrow{H_{\mathrm{f}}} & M_g\left(K[\![t]\!]\right) \\ \left(x_1(t), \dots, x_g(t)\right) & \longmapsto & \left(f_{ij}\left(x_i(t)\right)\right)_{ij}. \end{array}$$

Given $\mathrm{f} \in M_g\left(K[\![t]\!]\right)$ and $G = (G_1, \dots, G_g) \in K[\![t]\!]^g$, we consider the following differential equation in $X = (x_1, \dots, x_g)$,

$$H_{\mathrm{f}} \circ X \cdot X' = G. \tag{2}$$

We will always look for solutions of (2) in $\left(tK[\![t]\!]\right)^g$ in order to ensure that $H_{\mathrm{f}} \circ X$ is well defined. We further assume that $H_{\mathrm{f}}(0)$ is invertible in $M_g(K)$.

The next proposition guarantees the existence and the uniqueness of a solution of the differential equation (2).

Proposition 1. *Assuming that $H_{\mathrm{f}}(0)$ is invertible in $M_g(K)$, the system of differential equations (2) admits a unique solution in $K[\![t]\!]^g$.*

Proof. We are looking for a vector $X(t) = \sum_{n=1}^{\infty} X_n t^n$ that satisfies Equation (2). Since $X(0) = 0$ and $H_{\mathrm{f}}(0)$ is invertible in $K[\![t]\!]^g$, then $H_{\mathrm{f}}\big(X(t)\big)$ is invertible in $M_g\big(K[\![t]\!]\big)$. So Equation (2) can be written as

$$X'(t) = \big(H_{\mathrm{f}}(X(t))\big)^{-1} \cdot G(t). \tag{3}$$

Equation (3) applied to 0, gives the non-zero vector X_1. Taking the n-derivative of Equation (3) with respect to t and applying the result to 0, we observe that the coefficient X_n only appears on the hand left side of the result, so each component of X_n is a polynomial in the components of the X_i's for $i < n$ with coefficients in K. Therefore, the coefficients X_n exist and are all uniquely determined. □

We construct the solution of Equation (2) using a Newton scheme. We recall that for $Y = (y_1, \dots, y_g) \in K[\![t]\!]^g$, the differential of H_{f} with respect to Y is the function

$$\begin{array}{rccl} dH_{\mathrm{f}}(Y) \ : & K[\![t]\!]^g & \longrightarrow & M_g\big(K[\![t]\!]\big) \\ & h & \longmapsto & dH_{\mathrm{f}}(Y)(h) = \left(f'_{ij}\left(y_i\right) \cdot h_i\right)_{1 \le i,j \le g}. \end{array} \tag{4}$$

We fix $m \in \mathbb{N}$ and we consider an approximation X_m of X modulo t^m. We want to find a vector $h \in \left(t^m K[\![t]\!]\right)^g$, such that $X_m + h$ is a better approximation of X. We compute

$$H_{\mathrm{f}}\left(X_m + h\right) = H_{\mathrm{f}}\left(X_m\right) + dH_{\mathrm{f}}(X_m)(h) \pmod{t^{2m}}.$$

Therefore we obtain the following relation

$$\begin{aligned} H_{\mathrm{f}}\left(X_m + h\right) \cdot \left(X_m + h\right)' - G = \\ H_{\mathrm{f}}\left(X_m\right) \cdot X'_m + H_{\mathrm{f}}\left(X_m\right) \cdot h' + dH_{\mathrm{f}}(X_m)(h) \cdot X'_m - G \pmod{t^{2m-1}}. \end{aligned}$$

So we look for h such that

$$H_{\mathrm{f}}\left(X_m\right) \cdot h' + dH_{\mathrm{f}}(X_m)(h) \cdot X'_m = -H_{\mathrm{f}}\left(X_m\right) \cdot X'_m + G \pmod{t^{2m-1}}. \tag{5}$$

It is easy to see that the left hand side of Equation (5) is equal to $\left(H_{\mathrm{f}}\left(X_m\right) \cdot h\right)'$, therefore integrating each component of Equation (5) and multiplying the result by $\left(H_{\mathrm{f}}\left(X_m\right)\right)^{-1}$ gives the following expression for h

$$h = \left(H_{\mathrm{f}}\left(X_m\right)\right)^{-1} \int \left(G - H_{\mathrm{f}}\left(X_m\right) \cdot X'_m\right) dt \pmod{t^{2m}}, \tag{6}$$

where $\int Y\,dt$, for $Y \in K[\![t]\!]^g$, denotes the unique vector $I \in K[\![t]\!]^g$ such that $I' = Y$ and $I(0) = 0$. This formula defines a Newton operator for computing an approximation of the solution of Equation (2). Reversing the above calculations leads to the following proposition.

Proposition 2. *We assume that $H_{\mathrm{f}}(0)$ is invertible in $M_g(K)$. Let $m \geq 0$ be an integer, $n = 2m+1$ and $X_m \in K[\![t]\!]^g$ a solution of Equation (2) mod t^{m+1}. Then,*

$$X_n = X_m + \left(H_{\mathrm{f}}\left(X_m\right)\right)^{-1} \int \left(G - H_{\mathrm{f}}\left(X_m\right) \cdot X_m'\right)\, dt$$

is a solution of Equation (2) mod t^{n+1}.

It is straightforward to turn Proposition 2 into an algorithm that solves the non-linear system (2). We make a small optimization by integrating the computation of $H_{\mathrm{f}}(X)^{-1}$ in the Newton scheme.

Algorithm 1: Differential Equation Solver

`DiffSolve` (G, f, n)

Input : $G, \mathrm{f} \mod t^n$ such that $H_{\mathrm{f}}(0)$ is invertible in $M_g(K)$.

Output: The solution X of Equation (2) $\mod t^{n+1}$, $H_{\mathrm{f}}(X) \mod t^{\lceil n/2 \rceil}$

if $n = 0$ **then**
  return $0 \mod t$, $H_{\mathrm{f}}(0)^{-1} \mod t$

$m := \lceil \frac{n-1}{2} \rceil$;

$X_m, H_m :=$ `DiffSolve`(G, f, m);

$H_n := 2H_m - H_m \cdot H_{\mathrm{f}}(X) \cdot H_m \mod t^{m+1}$

return $X_m + H_n \int \left(G - H_{\mathrm{f}}\left(X_m\right) \cdot X_m'\right)\, dt \mod t^{n+1}$

According to Proposition 2, Algorithm 1 runs correctly when its entries are given with an infinite p-adic precision; however it could stop working if we use the fixed point arithmetic model. The next theorem guarantees its correctness in this type of model.

Theorem 3. *Let $n, g \in \mathbb{N}$, $N \in \frac{1}{e}\mathbb{Z}^*$, $G \in \mathcal{O}_K[\![t]\!]^g$ and $\mathrm{f} \in M_g\left(\mathcal{O}_K[\![t]\!]\right)$. We assume that $H_{\mathrm{f}}(0)$ is invertible in $M_g\left(\mathcal{O}_K\right)$ and that the components of the solution of Equation (2) have coefficients in $\mathcal{O}_K$. Then, the procedure* `DiffSolve` *runs with fixed point arithmetic at precision $O(p^M)$, with $M = \max(N, 3) + \lfloor \log_p(n) \rfloor$ if $p = 2$, $M = \max(N, 2) + \lfloor \log_p(n) \rfloor$ if $p = 3$ and $M = N + \lfloor \log_p(n) \rfloor$ otherwise, all the computations are done in $\mathcal{O}_K$ and the result is correct at precision $O(p^N)$.*

We give a proof of Theorem 3 at the end of Section 2.2. Right now, we concentrate on the complexity of Algorithm 1. Recall that $\mathrm{MM}(g, n)$ is the number of arithmetical operations required to compute the product of two $g \times g$ matrices containing polynomials of degree n and $\mathrm{M}(n) := \mathrm{MM}(1, n)$, therefore $\mathrm{M}(n)$ is the number of arithmetical operations required to compute the product of two polynomials of degree n. According to [22, Chapter 8], the two functions M(.) and MM$(g, .)$ (in the worst case) are related by the following formula

$$\mathrm{MM}(g, n) = O\left(g^\omega \mathrm{M}(n)\right) \tag{7}$$

where $\omega \in [2,3[$ is the exponent of matrix multiplication. Furthermore, we recall that $\mathrm{C}_H(n)$ denotes the algebraic complexity for computing $H \circ X \mod t^n$ for an analytic map $H : K[\![t]\!]^g \to M_g\left(K[\![t]\!]\right)$ of the form $H = H_{\mathrm{f}}$ where $\mathrm{f} \in M_g\left(K[\![t]\!]\right)$. We assume that $\mathrm{M}(n)$ and $\mathrm{C}_H(n)$ satisfy the superadditivity hypothesis

$$\begin{array}{rcl} \mathrm{M}(n_1+n_2) & \geq & \mathrm{M}(n_1)+\mathrm{M}(n_2), \\ \mathrm{C}_H(n_1+n_2) & \geq & \mathrm{C}_H(n_1)+\mathrm{C}_H(n_2), \end{array} \tag{8}$$

for all $n_1, n_2 \in \mathbb{N}$.
Using Equation (7) we deduce the following relation

$$O(\mathrm{MM}(g, n_1+n_2)) \quad \geq \quad \mathrm{MM}(g,n_1)+\mathrm{MM}(g,n_2). \tag{9}$$

Proposition 4. *Algorithm 1 performs* $O\left(\mathrm{MM}(g,n)+\mathrm{C}_{H_{\mathrm{f}}}(n)\right)$ *operations in* K.

Proof. Let D denote the algebraic complexity of Algorithm 1, then we have the following relation

$$\mathrm{D}(n) \leq \mathrm{D}\left(\left\lceil \frac{n-1}{2} \right\rceil\right) + O\left(\mathrm{MM}(g,n)+\mathrm{C}_{H_{\mathrm{f}}}(n)\right).$$

Noticing that g does not change at each iteration and using Equations (8) and (9), we find $\mathrm{D}(n) = O\left(\mathrm{MM}(g,n)+\mathrm{C}_{H_{\mathrm{f}}}(n)\right)$ and the result is proved. □

Remark 1. If the map H_{f} includes random univariate rational fractions of radicals of constant degrees, the algebraic complexity $\mathrm{C}_{H_{\mathrm{f}}}(n)$ is equal to $O\left(g^2\mathrm{M}(n)\right)$. Standard algorithms allow us to take $\mathrm{M}(n) \in \tilde{O}(n)$. Therefore, Algorithm 1 outputs the solution of Equation (2) mod t^{n+1} for a cost of $\tilde{O}(g^{\omega}n)$ operations in $\mathcal{O}_K$.

Corollary 5. *When performed with fixed point arithmetic at precision* $O(p^M)$*, the bit complexity of Algorithm 1 is* $O\left(\left(\mathrm{MM}(g,n)+\mathrm{C}_{H_{\mathrm{f}}}(n)\right)\cdot \mathrm{A}(K;M)\right)$ *where* $\mathrm{A}(K;M)$ *denotes an upper bound on the bit complexity of the arithmetic operations in* $\mathcal{O}_K/\pi^{eM}\mathcal{O}_K$.

2.2. Precision analysis

The goal of this subsection is to prove Theorem 3. The proof relies on the the theory of "differential precision" developed in [23, 24].
We study the solution $X(t)$ of Equation (2) when $G(t)$ varies, with the assumption $H_{\mathrm{f}}(0)$ is invertible in $M_g\left(\mathcal{O}_K\right)$. Proposition 1 showed that Equation (2) has a unique solution $X(G) \in K[\![t]\!]^g$. Moreover, if we examine the proof of Proposition 1, we see that the $n+1$ first coefficients of the vector $X(G)$ depends only on the first n coefficients of G. This gives a well-defined function

$$\begin{array}{rccc} X_n : & \left(K[\![t]\!]/\left(t^n\right)\right)^g & \longrightarrow & \left(tK[\![t]\!]/\left(t^{n+1}\right)\right)^g \\ & G & \longmapsto & X(G) \end{array}$$

for a given positive integer n. In addition, the proof of Proposition 1 states that for $G \in \left(K[\![t]\!]/\left(t^n\right)\right)^g$, $X_n(G)$ can be expressed as a polynomial in $G(0), G'(0), \ldots, G^{(n-1)}(0)$ with coefficients in K, therefore X_n is locally analytic.

Proposition 6. *For $G \in \left(K[\![t]\!]/\left(t^n\right)\right)^g$, the differential of X_n with respect to G is the following function*

$$\begin{aligned} dX_n(G) \ : \ \left(K[\![t]\!]/\left(t^n\right)\right)^g &\longrightarrow \left(tK[\![t]\!]/\left(t^{n+1}\right)\right)^g \\ \delta G &\longmapsto \left(H_{\mathrm{f}}\left(X_n(G)\right)\right)^{-1} \cdot \int \delta G. \end{aligned}$$

Proof. We differentiate the equation $H_{\mathrm{f}}\left(X_n(G)\right) \cdot X_n(G)' = G$ with respect to G. We obtain the following relation

$$H_{\mathrm{f}}\left(X_n(G)\right) \cdot \left(dX_n(G)(\delta G)\right)' + dH_{\mathrm{f}}\left(X_n(G)\right)\left(dX_n(G)(\delta G)\right) \cdot X_n(G)' = \delta G \tag{10}$$

where $dH_{\mathrm{f}}\left(X_n(G)\right)$ is the differential of H_{f} at $X_n(G)$ defined in (4). Making use of the relation

$$\left(\left(H_{\mathrm{f}}\left(X_n(G)\right)\right) \cdot dX_n(G)(\delta G)\right)' = H_{\mathrm{f}}\left(X_n(G)\right) \cdot \left(dX_n(G)(\delta G)\right)' + dH_{\mathrm{f}}\left(X_n(G)\right)\left(dX_n(G)(\delta G)\right) \cdot X_n(G)',$$

Equation (10) becomes

$$\left(H_{\mathrm{f}}\left(X_n(G)\right) \cdot dX_n(G)(\delta G)\right)' = \delta G.$$

Integrating the above relation and multiplying by $\left(H_{\mathrm{f}}\left(X_n(G)\right)\right)^{-1}$ we get the result. □

We now introduce some norms on $\left(K[\![t]\!]/\left(t^n\right)\right)^g$ and $\left(tK[\![t]\!]/\left(t^n\right)\right)^g$. We set $E_n = \left(K[\![t]\!]/\left(t^n\right)\right)^g$ and $F_n = \left(tK[\![t]\!]/\left(t^{n+1}\right)\right)^g$; for instance, X_n is a function from E_n to F_n.
First, we equip the vector space $K_n := K[\![t]\!]/\left(t^n\right)$ with the usual Gauss norm

$$\|a_0 + a_1 t + \cdots + a_{n-1}t^{n-1}\|_{K_n} = \max\left(\left|a_0\right|, \left|a_1\right|, \ldots, \left|a_{n-1}\right|\right).$$

We endow F_n with the norm obtained by the restriction of the induced norm $\|.\|$ on F_n: for every $X(t) = \left(x_i(t)\right)_i \in F_n$,

$$\|X(t)\|_{F_n} = \max_i \left\|x_i(t)\right\|_{K_n}.$$

On the other hand, we endow E_n with the following norm: for every $X(t) = \left(x_i(t)\right)_i \in E_n$,

$$\|X(t)\|_{E_n} = \left\|\int X(t)\right\|_{F_n} = \max_i \left\|\int x_i(t)\right\|_{K_n}.$$

Lemma 7. *Let $A \in M_g\left(\mathcal{O}_K[\![t]\!]/(t^n)\right)$. If there exists a vector $x(t) \in \left(\mathcal{O}_K[\![t]\!]/\left(t^n\right)\right)^g$ such that $\|A\,x\|_{F_n} < 1$ then A is not invertible in $M_g\left(\mathcal{O}_K[\![t]\!]/(t^n)\right)$.*

Proof. Write $A = (a_{ij}(t))_{i,j}$ and $x(t) = (x_1(t), \ldots, x_g(t))$. By definition, the norm $\|A\,x\|_{F_n}$ is equal to

$$\|A\,x\|_{F_n} = \max_i \left\|\sum_j a_{ij}x_j\right\|_{K_n}.$$

Therefore, the condition $\|A\,x\|_{F_n} < 1$ is equivalent to the following inequality

$$\left\| \sum_j a_{ij} x_j \right\|_{K_n} < 1 \tag{11}$$

for all $i = 1, \dots, g$. Let k be the residue field of K. Equation (11) implies that $\sum_j a_{ij} x_j = 0$ in k. Hence, the reduction of A in $M_g\big(k[\![t]\!]/(t^n)\big)$ is not invertible and A is not invertible in $M_g\big(\mathcal{O}_K[\![t]\!]/(t^n)\big)$. □

Lemma 8. *Let $G \in \big(\mathcal{O}_K[\![t]\!]/(t^n)\big)^g$. We assume that $X_n(G) \in \big(t\mathcal{O}_K[\![t]\!]/(t^n)\big)^g$, then $dX_n(G) : E_n \longrightarrow F_n$ is an isometry.*

Proof. The assumptions $X_n(G) \in \big(t\mathcal{O}_K[\![t]\!]/(t^n)\big)^g$ and $H_{\mathrm{f}}(0) \in \mathrm{GL}_g\big(\mathcal{O}_K\big)$ guarantee the invertibility of $H_{\mathrm{f}}\big(X_n(G)\big)$ in $M_g\big(\mathcal{O}_K[\![t]\!]\big)$. Let $\delta G \in E_n$ such that $\|\delta G\|_{E_n} = 1$. Using the fact that $H(X(G))^{-1} \int \delta G \in \big(t\mathcal{O}_K[\![t]\!]/(t^n)\big)^g$ and applying Lemma 7, we get

$$\|dX_n(G)(\delta G)\| = \|H(X(G))^{-1} \int \delta G\|_{F_n} = 1.$$

□

We define the following function:

$$\begin{array}{rccl} \tau_n & : & F_n \times E_n & \longrightarrow & \mathrm{Hom}(E_n, F_n) \\ & & (X\,,\,G) & \longmapsto & \left(\delta G \mapsto \big(H_{\mathrm{f}}(X)\big)^{-1} \cdot \int \delta G\right). \end{array}$$

By Proposition 6, the map dX_n is equal to $\tau_n \circ (X_n, \mathrm{id})$, where id denotes the identity map on E_n.

Lemma 9. *Let $x \in \mathbb{R}$ such that $x < -2\dfrac{\log p}{p-1}$, then $\Lambda\big(X_n\big)_{\geq 2}(x) < x$.*

Proof. One checks easily that $\Lambda(\mathrm{id})(x) = x$ and, by Lemma 9, $\Lambda(\tau_n)(x) \geq 0$ for all $x \in \mathbb{R}_+^*$. Applying [24, Proposition 2.5], we get

$$\Lambda\big(X_n\big)_{\geq 2}(x) \leq 2\left(x + \frac{\log p}{p-1}\right)$$

for all $x \leq -\dfrac{\log p}{p-1}$. Therefore, $\Lambda\big(X_n\big)_{\geq 2}(x) < x$ if $x < -2\dfrac{\log p}{p-1}$. □

Proposition 10. *Let $B_{E_n}(\delta)$ (resp. $B_{F_n}(\delta)$) be the closed ball in E_n (resp. in F_n) of center 0 and radius δ. Under the assumption of Lemma 8, we have for all $\delta < p^{\frac{-2}{p-1}}$,*

$$X_n\big(G + B_{E_n}(\delta)\big) = X_n(G) + B_{F_n}(\delta)\,.$$

Proof. As a direct consequence of [23, Proposition 3.12] and Lemma 9, we have the following formula

$$X_n\big(G + B_{E_n}(\delta)\big) = X_n(G) + dX_n(G)\big(B_{E_n}(\delta)\big),$$

for all $\delta < p^{\frac{-2}{p-1}}$. The result follows from Lemma 8. □

We end this section by giving a proof of Theorem 3.

Correctness proof of Theorem 3. Let G, f and n be the input of Algorithm 1. We first prove by induction on $n \geq 1$ the following equation

$$H_{\mathrm{f}}\big(X_n\big) \cdot X_n' = G \mod (t^n, p^M).$$

Let m be a positive integer and $n = 2m + 1$. Let $e_m = G - H_{\mathrm{f}}\big(X_m\big) \cdot X_m'$. From the relation

$$X_n = X_m + \big(H_{\mathrm{f}}\big(X_m\big)\big)^{-1} \int e_m \, dt \mod (t^{n+1}, p^M),$$

we derive the two formulas

$$H_{\mathrm{f}}\big(X_m\big) \cdot X_n = H_{\mathrm{f}}\big(X_m\big) \cdot X_m + \int e_m \, dt \mod (t^{n+1}, p^M) \tag{12}$$

and

$$\begin{aligned} H_{\mathrm{f}}\big(X_m\big) \cdot X_n' &= H_{\mathrm{f}}\big(X_m\big) \cdot X_m' + \big(H_{\mathrm{f}}\big(X_m\big)\big)' \cdot \big(X_m - X_n\big) + e_m \mod (t^n, p^M) \\ &= G + \big(H_{\mathrm{f}}\big(X_m\big)\big)' \cdot \big(X_m - X_n\big) \mod (t^n, p^M) \\ &= G - \big(H_{\mathrm{f}}\big(X_m\big)\big)' \cdot \big(H_{\mathrm{f}}\big(X_m\big)\big)^{-1} \int e_m \, dt \mod (t^n, p^M). \end{aligned}$$

Using the fact that the first m coefficients of e_m vanish, we get

$$H_{\mathrm{f}}\big(X_n\big) \cdot X_n' = H_{\mathrm{f}}\big(X_m\big) \cdot X_n' + dH_{\mathrm{f}}\big(X_m\big)\left(\big(H_{\mathrm{f}}\big(X_m\big)\big)^{-1} \int e_m \, dt\right) \cdot X_m' \mod (t^n, p^M). \tag{13}$$

In addition, one can easily verifies

$$dH_{\mathrm{f}}\big(X_m\big)\left(\big(H_{\mathrm{f}}\big(X_m\big)\big)^{-1} \int e_m \, dt\right) \cdot X_m' = \big(H_{\mathrm{f}}\big(X_m\big)\big)' \cdot \big(H_{\mathrm{f}}\big(X_m\big)\big)^{-1} \int e_m \, dt$$

Hence, Equation (13) becomes

$$H_{\mathrm{f}}\big(X_n\big) \cdot X_n' = G \mod (t^n, p^M).$$

Now, we define $G_n = H_{\mathrm{f}}\big(X_n\big) \cdot X_n'$ so that we have $X_n = X_n\big(G_n\big)$ and $\|G - G_n\|_{F_n} \leq p^{-M}$. Therefore, $\|G - G_n\|_{E_n} \leq p^{-M + \lfloor \log_p(n) \rfloor}$. By Proposition 10, we have that

$$X_n\big(G_n\big) = X_n(G) \mod (t^{n+1}, p^N).$$

Thus $X_n = X_n(G) \mod (t^{n+1}, p^N)$. □

3. Jacobians of curves and their isogenies

Throughout this section, the letter k refers to a fixed field of characteristic different from two. Let $\bar{k}$ be a fixed algebraic closure of k. In Section 3.1, we briefly recall some basic elements about principally polarized abelian varieties and $(\ell, \dots, \ell)$-isogenies between them; the notion of rational representation is discussed in Section 3.2. Finally, for a given rational representation, we construct a system of differential equations that we associate with it.

3.1. $(\ell, \cdots, \ell)$-isogenies between abelian varieties

Let A be an abelian variety of dimension g over k and $A^\vee$ be its dual. To a fixed line bundle $\mathcal{L}$ on A, we associate the morphism $\lambda_{\mathcal{L}}$ defined as follows

$$\begin{array}{rccc} \lambda_{\mathcal{L}} & : & A & \longrightarrow & A^\vee \\ & & x & \longmapsto & t_x^*\mathcal{L} \otimes \mathcal{L}^{-1} \end{array}$$

where t_x denotes the translation by x and $t_x^*\mathcal{L}$ is the pullback of $\mathcal{L}$ by t_x.
We recall from [25] that an *isogeny* between two abelian varieties is a surjective homomorphism of abelian varieties of finite kernel. The *degree* of an isogeny is the number of preimages of a generic point in its codomain.
A *polarization* λ of A is an isogeny $\lambda : A \longrightarrow A^\vee$, such that over $\bar{k}$, λ is of the form $\lambda_{\mathcal{L}}$ for some ample line bundle $\mathcal{L}$ on $A_{\bar{k}} := A \otimes \mathrm{Spec}(\bar{k})$. When the degree of a polarization λ of A is equal to 1, we say that λ is a *principal polarization* and the pair (A, λ) is a *principally polarized abelian variety*. We assume in the rest of this subsection that we are given a principally polarized abelian variety (A, λ). The *Rosati involution* on the ring $\mathrm{End}(A)$ of endomorphsims of A corresponding to the polarization λ is the map

$$\begin{array}{ccc} \mathrm{End}(A) & \longrightarrow & \mathrm{End}(A) \\ \alpha & \longmapsto & \lambda^{-1} \circ \alpha^\vee \circ \lambda. \end{array}$$

The Rosati involution is crucial for the study of the division algebra $\mathrm{End}(A) \otimes \mathbb{Q}$, but for our purpose, we only state the following result.

Proposition 11. *[25, Proposition 14.2] For every $\alpha \in End(A)$ fixed by the Rosati involution, there exists, up to algebraic equivalence, a unique line bundle $\mathcal{L}_A^\alpha$ on A such that $\lambda_{\mathcal{L}_A^\alpha} = \lambda \circ \alpha$. In particular, taking α to be the identity endomorphism denoted "1", there exists a unique line bundle $\mathcal{L}_A^1$ such that $\lambda_{\mathcal{L}_A^1} = \lambda$.*

The notion of *algebraic equivalence* is defined as follows. We say that two line bundles $\mathcal{L}_1$ and $\mathcal{L}_2$ on A are algebraically equivalent if they can be connected by a third line bundle, *i.e.* if there exist a connected scheme X, two closed points $x_1, x_2 \in X$ and a line bundle $\mathcal{N}$ on $A \times X$, such that $\mathcal{N}_{|A\times\{x_1\}} \simeq \mathcal{L}_1$ and $\mathcal{N}_{|A\times\{x_2\}} \simeq \mathcal{L}_2$. We say that two divisors on A are algebraically equivalent if their corresponding line bundles are.

Using Proposition 11, we give the definition of an $(\ell, \dots, \ell)$-isogeny.

Definition 12. *Let (A_1, λ_1) and (A_2, λ_2) be two principally polarized abelian varieties of dimension g over k and $\ell \in \mathbb{N}^*$. An $(\ell, \dots, \ell)$-isogeny I between A_1 and A_2 is an isogeny $I : A_1 \longrightarrow A_2$ such that*

$$I^*\mathcal{L}_{A_2}^1 = \mathcal{L}_{A_1}^\ell,$$

where $\mathcal{L}^{\ell}_{A_1}$ is the unique line bundle on A_1 associated with the multiplication by ℓ map.

We now suppose that A is the Jacobian of a genus g curve C over k. We will always make the assumption that there is at least one k-rational point on C. Let r be a positive integer and fix $P \in C$. We define $C^{(r)}$ to be the symmetric power of C and $j_P^{(r)}$ to be the map

$$\begin{array}{rcl} C^{(r)} & \longrightarrow & A \simeq J(C) \\ (P_1, \ldots, P_r) & \longmapsto & [P_1 + \cdots P_r - rP]. \end{array}$$

If $r = 1$ then the map $j_P^{(1)}$ is called the *Jacobi map* with origin P. We write j_P for the map $j_P^{(1)}$. The image of $j_P^{(r)}$ is a closed subvariety of A which can be also written as r summands of $j_P(C)$. Let Θ be the image of $j_P^{(g-1)}$, it is a divisor on A and when P is replaced by another point, Θ is replaced by a translate. We call Θ the theta divisor associated with A.

Remark 2. If A is the Jacobian of a curve C and Θ its theta divisor, then $\mathcal{L}^1_A = \mathcal{L}(\Theta)$, where $\mathcal{L}(\Theta)$ is the sheaf associated to the divisor Θ.

Using Remark 2, Definition 12 for Jacobian varieties gives the following

Proposition 13. *Let $\ell \in \mathbb{N}^*$, A_1 and A_2 be the Jacobians of two algebraic curves over k and Θ_1 and Θ_2 be the theta divisors associated to A_1 and A_2 respectively. If an isogeny $I : A_1 \longrightarrow A_2$ is an $(\ell, \ldots, \ell)$-isogeny then $I^*\Theta_2$ is algebraically equivalent to $\ell\Theta_1$.*

Proof. For all $x \in A_1$, the theorem of squares [25, Theorem 5.5] gives the following relation

$$t^*_{\ell x}\,\mathcal{L}^1_{A_1} \otimes \left(\mathcal{L}^1_{A_1}\right)^{-1} = \left(t^*_x\,\mathcal{L}^1_{A_1} \otimes \left(\mathcal{L}^1_{A_1}\right)^{-1}\right)^{\otimes \ell} = t^*_x\left(\mathcal{L}^1_{A_1}\right)^{\otimes \ell} \otimes \left((\mathcal{L}^1_{A_1})^{\otimes \ell}\right)^{-1}.$$

Meaning that,

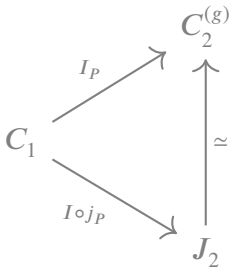
$$\lambda_{(\mathcal{L}^1_{A_1})^{\otimes \ell}} = \lambda_{\mathcal{L}^{\ell}_{A_1}}.$$

From Proposition 11, we deduce that the line bundle $\mathcal{L}^{\ell}_{A_1}$ is algebraically equivalent to $\left(\mathcal{L}^1_{A_1}\right)^{\otimes \ell}$, therefore $I^*\mathcal{L}^1_{A_2}$ and $\left(\mathcal{L}^1_{A_1}\right)^{\otimes \ell}$ are algebraically equivalent. By Remark 2, $I^*\mathcal{L}^1_{A_2}$ corresponds to $I^*\Theta_2$ and $\left(\mathcal{L}^1_{A_1}\right)^{\otimes \ell}$ corresponds to $\ell\Theta_1$. □

3.2. Rational representation of an isogeny between Jacobians of hyperelliptic curves

We focus on computing an isogeny between Jacobians of hyperelliptic curves. Let C_1 (resp. C_2) be a genus g hyperelliptic curve over k, J_1 (resp. J_2) be its associated Jacobian and Θ_1 (resp. Θ_2) be its theta divisor. We suppose that there exists a separable isogeny $I : J_1 \longrightarrow J_2$. Let $P \in C_1$ be a Weierstrass point, let $j_P : C_1 \longrightarrow J_1$ be the Jacobi map with origin P. Generalizing [14, Proposition 4.1] gives the following proposition

Proposition 14. *The morphism $I \circ j_P$ induces a unique morphism $I_P : C_1 \longrightarrow C_2^{(g)}$ such that the following diagram commutes*



We assume that C_1 (resp. C_2) is given by the singular model $v^2 = f_1(u)$ (resp. $y^2 = f_2(x)$), where f_1 (resp. f_2) is a polynomial of degree $2g+1$ or $2g+2$. Set $Q = (u, v) \in C_1$ and $I_P(Q) = \{(x_1, y_1), \dots, (x_g, y_g)\}$. We use the Mumford's coordinates to represent the element $I_P(Q)$: it is given by a pair of polynomials $(U(X), V(X))$ such that

$$U(X) = X^g + \sigma_1 X^{g-1} + \cdots + \sigma_g$$

where

$$\sigma_i = (-1)^i \sum_{1 \le j_1 < j_2 < \cdots < j_i \le g} x_{j_1} x_{j_2} \cdots x_{j_i}$$

and

$$V(X) = \rho_1 X^{g-1} + \cdots + \rho_g = \sum_{j=0}^{g-1} y_j \left(\prod_{i=0, i \neq j}^{g-1} \frac{X - x_i}{x_j - x_i} \right).$$

The tuple $(\sigma_1, \cdots, \sigma_g, \rho_1, \cdots, \rho_g)$ consists of rational functions (on C_1) in u and v and it is called a *rational representation* of I.
We recall that the *degree* of a rational function f on a curve C, denoted by $\deg(f)$, is the number of its zeros (or poles).

Lemma 15. *Let $\pi : C_1 \to \mathbb{P}^1$ be a rational function on C_1.*

1. *If $\pi(u, v)$ is invariant under the hyperelliptic involution of C_1 then there exists a rational fraction A in u such that*

   $$\pi(u, v) = A(u)$$

   and $\deg(A) \le \deg(\pi)/2$.

2. *Otherwise, we can always find two rational fractions B and D in u such that*

   $$\pi(u, v) = B(u) + vD(u)$$

   and the degrees of B and D are bounded by $\deg(\pi)$ *and* $\deg(\pi)+g+1$ *respectively. Moreover, if $B(u) = 0$ then* $\deg(D) \le \deg(\pi)/2 + g + 1$.

Proof. 1. The inequality $\deg(A) \le \deg(\pi)/2$ comes from the fact that the function u has degree 2.

2. The rational fractions $B(u)$ and $D(u)$ verify the following relations

$$B(u) = \frac{\pi(u,v) + \pi(u,-v)}{2}, \quad D(u) = \frac{\pi(u,v) - \pi(u,-v)}{2v}.$$

Since, $\pi(u,v)+\pi(u,-v)$ and $\pi(u,v)-\pi(u,-v)$ are invariant under the hyperelliptic involution and have degrees bounded by $2\deg(\pi)$, then $B(u)$ is a rational fraction of degree bounded by $\deg(\pi)$ and $D(u)$ is a rational fraction of degree bounded by $\deg(\pi)+g+1$ (Note that v is a rational fraction of degree bounded by $2g+2$).

□

Proposition 16. *The functions $\sigma_1, \dots, \sigma_g$ can be seen as rational fractions in u and have the same degree bounded by $\deg(\sigma_1)/2$. Moreover, the rational functions $\rho_1/v, \dots, \rho_g/v$ can also be expressed as rational fractions in u of degrees bounded by $\deg(\rho_1)/2+g+1, \dots, \deg(\rho_g)/2+g+1$ respectively.*

Proof. It is a direct consequence of Lemma 15 and using the fact that $I_P(u,-v) = -I_P(u,v)$. □

Remark 3. If P is not a Weierstrass point, there exists rational fractions A_i,B_i,D_i and E_i in u such that $\sigma_i(u,v) = A_i(u) + vB_i(u)$ and $\rho_i(u,v) = D_i(u) + vE_i(u)$ for all $i \in \{1, \cdots, g\}$. Let $\bar{P}$ the image of P by the hyperelliptic involution. The morphism $I_{\bar{P}}$ gives a rational representation $(\overline{\sigma_1}, \cdots, \overline{\sigma_g}, \overline{\rho_1}, \cdots, \overline{\rho_g})$ of I. From the relation $I_P(u,-v) = -I_{\bar{P}}(u,v)$, we deduce $\overline{\sigma_i}(u,v) = A_i(u) - vB_i(u)$ and $\overline{\rho_i}(u,v) = -D_i(u) + vE_i(u)$ for all $i \in \{1, \cdots, g\}$. This gives the following formulas

$$A_i(u) = (\sigma_i(u,v) + \overline{\sigma_i}(u,v))/2, \quad B_i(u) = (\sigma_i(u,v) - \overline{\sigma_i}(u,v))/2v,$$

$$D_i(u) = (\rho_i(u,v) - \overline{\rho_i}(u,v))/2, \quad E_i(u) = (\rho_i(u,v) + \overline{\rho_i}(u,v))/2v.$$

The degrees of A_i and D_i (resp. B_i and E_i) are bounded by $\deg(\sigma_i)$ (resp. $\deg(\rho_i)+g+1$).

In order to determine the isogeny I, it suffices to compute its rational representation (because I is a group homomorphism), so we need to have some bounds on the degrees of the rational functions $\sigma_1, \dots, \sigma_g, \rho_1, \dots, \rho_g$. In the case of an $(\ell, \dots, \ell)$-isogeny, we adapt the proof of [11, § 6.1] in order to obtain bounds in terms of ℓ and g.

Lemma 17. *Let $i \in \{1, \dots, g\}$. The pole divisor of σ_i seen as function on J_2, is algebraically equivalent to $2\Theta_2$. The pole divisor of ρ_i seen as function on J_2 is algebraically equivalent to $3\Theta_2$ if $\deg(f_2) = 2g+1$, and $4\Theta_2$ otherwise.*

Proof. This is a generalization of [14, Lemma 4.25]. Note that if $\deg(f_2) = 2g+1$, then σ_i has a pole of order two along the divisor $\{(R_1, \dots, R_{g-1}, \infty)\,;\, R_i \in C_2\}$ which is algebraically equivalent to Θ_2. □

Lemma 18 ([26, Appendix])**.** *The divisor $j_P(C_1)$ of J_1 is algebraically equivalent to $\dfrac{\Theta_1^{g-1}}{(g-1)!}$ where Θ_1^{g-1} denotes the $g-1$ times self intersection of the divisor Θ_1.*

Proposition 19. *Let ℓ be a non-zero positive integer and $i \in \{1, \dots, g\}$. If I is an $(\ell, \dots, \ell)$-isogeny, then the degree of σ_i seen as a function on C_1 is bounded by $2g\ell$. The degree of ρ_i seen as a function on C_1 is bounded by $3g\ell$ if $\deg(f_2) = 2g + 1$, and $4g\ell$ otherwise.*

Proof. The degrees of $\sigma_1, \dots, \sigma_g, \rho_1, \dots, \rho_g$ are obtained by computing the intersection of $I_P(C)$ with their pole divisors. By Lemma 17, it suffices to show that

$$I_P(C) \cdot \Theta_2 = \ell g.$$

Since I is an $(\ell, \dots, \ell)$-isogeny, Proposition 13 gives that $I^*\Theta_2$ is algebraically equivalent to $\ell\Theta_1$. Moreover, up to algebraic equivalence,

$$I^*\big(I_P(C)\big) = \big(|\ker(I)|\big)\, j_P(C) = \ell^g j_P(C).$$

Using Lemma 18, we obtain

$$I^*\big(I_P(C)\big) \cdot I^*\Theta_2 = g\ell^{g+1}.$$

As

$$I^*\big(I_P(C)\big) \cdot I^*\Theta_2 = \deg(I)\,\big(I_P(C) \cdot \Theta_2\big) = \ell^g (I_P(C) \cdot \Theta_2\big),$$

the result follows. □

3.3. Associated differential equation

We assume that $\mathrm{char}(k) \neq 2$. We generalize [11, § 6.2] by constructing a differential system modeling the map $F_P = I \circ j_P$ of Proposition 14. The map F_P is a morphism of varieties, it acts naturally on the spaces of holomorphic differentials $H^0(J_2, \Omega^1_{J_2})$ and $H^0(C_1, \Omega^1_{C_1})$ associated to J_2 and C_1 respectively, this action gives a map

$$F_P^* \ : \ H^0(J_2, \Omega^1_{J_2}) \longrightarrow H^0(C_1, \Omega^1_{C_1}).$$

A basis of $H^0(C_1, \Omega^1_{C_1})$ is given by

$$B_1 = \left\{ u^i \frac{du}{v} \, ; i \in \{0, \dots, g-1\} \right\}.$$

The Jacobi map of C_2 induces an isomorphism between the spaces of holomorphic differentials associated to C_2 and J_2, so $H^0(J_2, \Omega^1_{J_2})$ is of dimension g, it can be identified with the space $H^0(C_2^g, \Omega^1_{C_2^g})^{S_n}$ (here the symmetric group S_n acts naturally on the space $H^0(C_2^g, \Omega^1_{C_2^g})$). With this identification, a basis of $H^0(J_2, \Omega^1_{J_2})$ is chosen to be equal to

$$B_2 = \left\{ \sum_{j=1}^{g} x_j^i \frac{dx_j}{y_j} \, ; \, i \in \{0, \dots, g-1\} \right\}.$$

Let $(m_{ij})_{0 \leq i,j \leq g} \in \mathrm{GL}_g(\bar{k})$ be the matrix of F_P^* with respect of these two bases, we call it the *normalization matrix*.

Remark 4. Let P_1 and P_2 be two points on C_1. The two morphisms I_{P_1} and I_{P_2} satisfy the following relation

$$I_{P_1} = I_{P_2} + I([P_2 - P_1]).$$

Therefore, the linear maps $I^*_{P_1}$ and $I^*_{P_2}$ are equal.

Let $Q = (u_Q, v_Q) \in C_1$ be a non-Weierstrass point different from P and $I_P(Q) = \{R_1, \dots, R_g\}$ such that $I_P(Q)$ contains g distinct points and does not contain neither a point at infinity nor a Weierstrass point. The points R_i may be defined over an extension k' of k of degree equal to $O(g!)$. Let t be a formal parameter of C_1 at Q, then we have the following diagram

$$\begin{array}{ccc} \mathrm{Spec}\big(k'[\![t]\!]\big) & \xrightarrow{t\mapsto(R_i(t))_i} & C_2^g \\ \downarrow & & \downarrow \\ C_1 & \xrightarrow{I_P} & C_2^{(g)} \end{array}$$

For all $i = 1, \dots, g$, the pull back of $\sum_{j=1}^{i-1} x^{i-1} dx_j/y_j$ along the bottom horizontal arrow, then along the left vertical arrow, gives

$$\frac{du}{v} \sum_{j=1}^{g} m_{ij} u^{j-1}.$$

And the pull back of $\sum_{j=1}^{i-1} x^{i-1} dx_j/y_j$ along the right vertical arrow, then along the top horizontal arrow gives

$$\sum_{j=1}^{g} x_j^{i-1} dx_j.$$

This gives the differential system

$$\begin{cases} \dfrac{dx_1}{y_1} + \cdots + \dfrac{dx_g}{y_g} = \big(m_{11} + m_{12}\cdot u + \ldots + m_{1g}\cdot u^{g-1}\big)\dfrac{du}{v}, \\ \dfrac{x_1\cdot dx_1}{y_1} + \cdots + \dfrac{x_g\cdot dx_g}{y_g} = \big(m_{21} + m_{22}\cdot u + \ldots + m_{2g}\cdot u^{g-1}\big)\dfrac{du}{v}, \\ \qquad\vdots \qquad\qquad\qquad\qquad \vdots \\ \dfrac{x_1^{g-1}\cdot dx_1}{y_1} + \cdots + \dfrac{x_g^{g-1}\cdot dx_g}{y_g} = \big(m_{g1} + m_{g2}\cdot u + \ldots + m_{gg}\cdot u^{g-1}\big)\dfrac{du}{v}, \\ y_1^2 = f_2(x_1), \quad \cdots \quad , y_g^2 = f_2(x_g). \end{cases} \tag{14}$$

Equation (14) has been initially constructed and solved in [11] for $g = 2$. In this case, the normalization matrix and the initial condition $(x_1(0), x_2(0))$ are computed using algebraic theta functions. In a more practical way, we refer to [14] for an easy computation of the initial condition $(x_1(0), x_2(0))$ of Equation (14) and for solving the differential system using a Newton iteration. However, in this case, the normalization matrix is determined by differentiating modular equations. There is a slight difference in Equation (14) between the two cases, especially $x_1(0)$ and $x_2(0)$ are different in the first, and equal in the second. Let H be the g-squared matrix defined by

$$H(x_1, \dots x_g) = \left(x_j^{i-1} \frac{1}{y_j} \right)_{1 \le i,j \le g}.$$

We suppose that $g = 2$. If the initial condition $(x_1(0), x_2(0))$ of Equation (14) satisfies $x_1(0) \neq x_2(0)$, then the matrix $H(x_1(0), x_2(0))$ is invertible in $M_2(k')$. Otherwise, its determinant is equal to zero. More generally, we prove that with the assumptions that we made on $Q, R_1, R_2, \dots R_{g-1}$ and R_g, the matrix $H(x_1(0), \dots, x_g(0))$ is invertible in $M_g(k')$. Let t be a formal parameter, $Q(t) = (u(t), v(t))$ the formal point on $C_1\left(k[\![t]\!]\right)$ that corresponds to $t = u - u_Q$ and $\{R_1(t), \dots, R_g(t)\}$ the image of $Q(t)$ by I_P, then Equation (14) becomes

$$H(X(t)) \cdot X'(t) = G(t) \tag{15}$$

where $X(t) = (x_1(t), \dots, x_g(t))$ and $G(t) = v^{-1} \left(\sum_{i=1}^{g} m_{ij} u^{i-1} \right)_{1 \le j \le g}$. Thus we have the following proposition.

Proposition 20. *The matrix $H(X(t))$ is invertible in $M_g\left(k'[\![t]\!]\right)$.*

Proof. The matrix $H(X(t))$ is an alternant matrix, its determinant is given by

$$\det\left(H(X(t))\right) = \frac{\prod_{1 \le i < j \le g} \left(x_j(t) - x_i(t)\right)}{\prod_{i=1}^{g} y_i(t)}$$

which is invertible in $M_g\left(k'[\![t]\!]\right)$ because $x_i(0) \neq x_j(0)$ for all $i, j \in \{1, \dots, g\}$ such that $i \neq j$. □

Corollary 21. *Let p be a prime number. We assume that k is an extension of $\mathbb{Q}_p$. Up to a change of variables, Equation (15) fulfills all the assumptions of Equation (2), in particular it admits a unique solution in $k'[\![t]\!]$.*

By Corollary 21, it is straightforward to make use of Algorithm 1 to solve Equation (15) when k is an unramified extension of the field of p-adic numbers. This gives rise to an algorithm that computes a rational representation of a given $(\ell, \cdots, \ell)$-isogeny between Jacobians of hyprelliptic curves of genus g, whose complexity is quasi-optimal with respect to ℓ but not in g. Thus, Algorithm 1 can only be used efficiently to compute isogenies of Jacobians of hyperelliptic curves of small genus.

4. Solving alternant systems of differential equations

Let p be an odd prime number. In this section, we aim for effective resolution of Equation (15) when it is defined over an unramified extension of $\mathbb{Q}_p$. We re-examine the Newton scheme of Algorithm 1 to make it quasi-linear in the dimension of the solution $X(t)$. This will give a quasi-optimal algorithm to compute rational representations of isogenies between Jacobians of hyperelliptic curves over finite fields, after having possibly lifted the problem in the p-adics.

The three next subsections are concerned with preliminary material: we introduce the differential system that we want to solve, then we recall some computational results that will eventually be used in our main algorithm exposed in Section 4.4.

4.1. The setup

We keep the same notation as Section 3.3 and we assume that $p \neq 2$ and k is a finite field of characteristic p. For $i \in \{1, \dots, g\}$, write $R_i = (x_i^{(0)}, y_i^{(0)})$. We recall that the computation of the rational representation associated with I_P reduces to the problem of computing an approximation of the following differential system whose unknown is $X(t) = (x_1(t), \dots, x_g(t)) \in k'$.

$$\begin{cases} H(X(t)) \cdot X'(t) = G(t), \quad X(0) = (x_1^{(0)}, \cdots, x_g^{(0)}), \\ y_j(t)^2 = f_2(x_j(t)), \quad y_j(0) = y_j^{(0)} \quad \text{for } j = 1, \dots, g \end{cases} \tag{16}$$

where $G(t) \in k[\![t]\!]^g$ and $H(X(t))$ are the matrices defined by

$$G(t) = \frac{1}{v(t)} \left(\sum_{i=1}^{g} m_{ij} u(t)^{i-1} \right)_{1 \leq j \leq g} \quad \text{and} \quad H(X(t)) = \left(\frac{x_j(t)^{i-1}}{y_j(t)} \right)_{1 \leq i,j \leq g}.$$

Based on the discussion in Section 1.1, we are sometimes obliged to lift Equation (16) to the p-adics. Therefore, we will replace k by an unramified extension K_0 of $\mathbb{Q}_p$ and k' by an unramified extension K of K_0 of degree at most $O(g)$. Consequently, f_1, f_2 and the components of $G(t)$ have coefficients in $\mathcal{O}_{K_0}$; moreover, $X(t) \in \mathcal{O}_K[\![t]\!]^g$.

By Corollary 21, Equation (16) can be solved using the following Newton iteration

$$X_{2m+1}(t) = X_m(t) + H(X_m(t))^{-1} \int (G - H(X_m(t)) \cdot X_m'(t))\, dt.$$

Or, equivalently,

$$H(X_m(t)) \cdot (X_{2m+1}(t) - X_m(t)) = \int (G - H(X_m(t)) \cdot X_m'(t))\, dt. \tag{17}$$

A call from Algorithm 1 gives the desired result, but this is not optimal in g. As explained in Section 1.1, this lack of efficiency is due to the fact that the components of the solution $X(t)$ of Equation (16) have coefficients defined over the field K, whose degree over K_0 depends on g. For this reason, we work directly on the first Mumford polynomial

$$U(t, z) = \prod_{j=1}^{g} (z - x_j(t))$$

whose coefficients are defined over the ring $K_0[\![t]\!]$: we rewrite the Newton scheme (17) accordingly and design fast algorithms for iterating it in quasi-linear time.

4.2. Computing Newton sums

We recall an efficient algorithm for the computation of the Newton sums of a polynomial. Let K_0 be an unramified extension of $\mathbb{Q}_p$. Let $P(t,z)$ be a monic polynomial of degree d with coefficients in $K_0[\![t]\!]$ such that $P(0,z)$ is separable over K_0 and $x_1(t), x_2(t), \cdots, x_d(t)$ its roots in $K[\![t]\!]$, where K denotes the splitting field of $P(0,z)$. We define the *i-th Newton sum* $s_i(t)$ of P by

$$s_i(t) = \sum_{j=1}^{d} x_j(t)^i \in K_0[\![t]\!],$$

and we are interested in designing an efficient algorithm to compute it, only from the coefficients of P.

Let P^* be the reciprocal polynomial of P, *i.e.* $P^*(t,z) = \prod_{j=1}^{d}(1 - x_j(t)z)$. It is well known that the i-th coefficient of the power series expansion of $(P^*)'/P^*$ in $K_0[\![t]\!][\![z]\!]$ is equal to $-s_{i+1}(t)$ (see for instance [27, Lemma 2]). This gives Algorithm 2 to compute the first g Newton sums of the polynomial $P(t,z)$ modulo t^{n+1}.

Algorithm 2: Newton Sums

`NewtonSums` (P, g, n)

Input : $P \in K_0[\![t]\!][z]$, $g \in \mathbb{N}^*$, $n \in \mathbb{N}^*$.

Output: The sequence $s_1(t) \mod t^{n+1}, \cdots, s_g(t) \mod t^{n+1}$.

$f := -(P^*)'/P^* \mod (t^{n+1}, z^g)$; // $f = \sum_{i=0}^{g-1} f_i(t) z^i$

return $f_0(t), \cdots, f_{g-1}(t)$

Proposition 22. *Let $P \in \mathcal{O}_{K_0}[\![t]\!][z]$ be a monic polynomial and $g, n, N \in \mathbb{N}^*$. When the procedure* `NewtonSums` *runs with fixed point arithmetic at precision $O(p^N)$, all the computations are done in $\mathcal{O}_{K_0}$ and the result is correct at precision $O(p^N)$. Moreover, Algorithm 2 performs at most $\tilde{O}(ng)$ operations in K_0.*

Proof. The fact that all the computations stay within $\mathcal{O}_{K_0}$ is a direct consequence of the assumption $P \in \mathcal{O}_{K_0}[\![t]\!][z]$ and the fact that P^* is invertible in $\mathcal{O}_{K_0}[\![t]\!][\![z]\!]$. In addition, it is easy to see that the output of `NewtonSums`(P, g, n) is correct at precision $O(p^N)$.

The inverse power series of P^* modulo (t^{n+1}, z^g) in $K_0[\![t]\!][\![z]\!]$ is computed by the Newton iteration $Q \mapsto Q(2 - QP^*)$. Therefore, the complexity of the computation of f only depends on the complexity of multiplying two bivariate polynomials of total degree $n + g$. This can be done using at most $\tilde{O}(gn)$ operations in K_0. □

4.3. Hankel matrix-vector product

Let $g \in \mathbb{N}^*$. We recall that a $g \times g$ Hankel matrix A is a $g \times g$ matrix of the form

$$A = \begin{pmatrix} a_0 & a_1 & a_2 & \cdots & \cdots & a_{g-1} \\ a_1 & a_2 & & \cdots & \cdots & a_g \\ a_2 & & & & & \vdots \\ \vdots & & & & & \vdots \\ a_{g-1} & \cdots & \cdots & a_{2g-4} & a_{2g-3} & a_{2g-2} \end{pmatrix}.$$

Matrix-vector multiplication for this type of matrix can be computed in $O(M(g))$ arithmetic operations instead of $O(g^\omega)$, where $\omega \in]2, 3]$ is a feasible exponent of matrix multiplication.

Proposition 23. *Let $n \in \mathbb{N}$. Let K_0 be an unramified extension of $\mathbb{Q}_p$. Let $A = (a_{i+j-2}(t))_{i,j} \in M_g(K_0[\![t]\!])$ be a Hankel matrix and $v = (v_1(t), \cdots, v_g(t)) \in K_0[\![t]\!]^g$. Let f and h be the two polynomials*

$$f(t, z) = a_0(t) + a_1(t)z + a_2(t)z^2 + \cdots + a_{2g-3}(t)z^{2g-3} + a_{2g-2}(t)z^{2g-2}$$

and

$$h(t, z) = v_1(t)z^{g-1} + v_2(t)z^{g-2} + \cdots + v_g(t).$$

Write $f(t, z) \cdot h(t, z) \mod (t^{n+1}, z^{2g-1}) = \sum_{i=0}^{2g-2} w_i(t)z^i$ then

$$A \cdot v \mod t^{n+1} = (w_{g-1}, \cdots, w_{2g-2}).$$

Proof. Let $i \in \{1, \ldots, g\}$. The $(g-2+i)$-th coefficient of the product $R = f(t, z) \cdot h(t, z)$ is equal to

$$R_{g-2+i}(t) = \sum_{j=i-1}^{g-2+i} a_j(t)v_{j+2-i}(t) = \sum_{j=1}^{g} a_{i+j-2}(t)v_j(t).$$

Therefore, $R_{g-2+i}(t)$ is the i-th component of the product $A \cdot v$. □

Proposition 23 gives a quasi-linear algorithm to compute the matrix-vector product for Hankel matrices.

Proposition 24. *Let $n \in \mathbb{N}$. Let K_0 be an unramified extension of $\mathbb{Q}_p$. Let $A = (a_{i+j-2}(t))_{i,j} \in M_g(K_0[\![t]\!])$ be a Hankel matrix and $v = (v_1(t), \cdots, v_g(t)) \in K_0[\![t]\!]^g$. When it is called on the input (A, v, n), the algorithm* `HankelProd` *performs at most $\tilde{O}(ng)$ operations in K_0.*

Proof. This is a direct consequence of Proposition 23. □

Algorithm 3: Hankel matrix-vector product

`HankelProd` (A, v, n)

Input : $A = (a_{i+j-2}(t))_{i,j} \mod t^{n+1}$, $v = (v_1(t), \cdots, v_g(t)) \mod t^{n+1}$, $n \in \mathbb{N}$.

Output: The product $A \cdot v \mod t^{n+1}$.

$f := a_0 + a_1 z + a_2 z^2 + \cdots + a_{2g-3} z^{2g-3} + a_{2g-2} z^{2g-2}$;

$h := v_1 z^{g-1} + v_2 z^{g-2} + \cdots + v_g$;

$w := fh \mod (t^{n+1}, z^{2g-1})$; // $w = \sum_{i=0}^{2g-2} w_i z^i$

return $w_{g-1}, \cdots, w_{2g-2}$

4.4. The alternant differential system

We go back to the system of differential equations (16). We will make use of Equation (17) to construct its solution by successive approximations.

Let $m \geq 0$ be an integer and $n = 2m + 1$. We suppose that we are given an approximation $U_m(t, z) = \prod_{j=1}^{g} (z - x_j^{(m)}(t)) \in K_0[\![t]\!][z]$ of the polynomial $U(t, z)$ modulo t^{m+1}, such that the vector $X_m = (x_1^{(m)}(t), \ldots x_g^{(m)}(t))$ satisfies Equation (16) modulo t^m. In order to compute an approximation $U_n(t, z)$ of $U(t, z)$ modulo t^{n+1} from $U_m(t, z)$, we use Equation (17) and perform the three following steps:

1. Compute $H(X_m(t)) \cdot X'_m(t)$ modulo t^{n+1}.
2. Compute $F_m(t) = \int (G - H(X_m(t)) \cdot X'_m(t))\, dt. \mod t^{n+1}$
3. Compute U_n by solving linear system $H(X_m(t)) \cdot (X_n(t) - X_m(t)) = F_m(t)$ modulo t^{n+1}.

Once step 1 is carried out, step 2 can be executed using at most $\tilde{O}(ng)$ operations in K_0, because the components of the vector $H(X_m(t)) \cdot X'_m(t)$ are defined over K_0. The following construction will show that the vector X_m will not be of any use to perform steps 1 and 3 but only the polynomial $U_m(t, z)$.

Write $f_2(z) = \sum_{j=1}^{g-1} f_j z^j$. Let $s_i^{(m)}(t) \in K_0[\![t]\!]$ be the i-th Newton sum of $U_m(t, z)$ and

$$r_i^{(m)}(t) = \frac{1}{i} \frac{d\, s_i^{(m)}(t)}{dt} = \sum_{j=1}^{g} \frac{d\, x_j^{(m)}(t)}{dt} \cdot x_j^{(m)}(t)^{i-1} \in K_0[\![t]\!]. \tag{18}$$

Let $W_m(t, z) = \sum_{i=0}^{g-1} w_i^{(m)}(t)\, z^i \in K_0[\![t]\!][z]$ be the degree $g - 1$ polynomial such that

$$W_m(t, z)^2 = \frac{1}{f_2(z)} \mod (t^{n+1}, U_m(t, z))$$

with the initial condition $W_m(0, x_j^{(0)}) = y_j^{(0)}$, for all $j \in \{1, \dots, g\}$.
Set $V_m(t, z) = f_2(z) \cdot W_m(t, z) \mod (t^{n+1}, U_m(t, z))$ and, for $j \in \{1, \dots, g\}$, let $y_j^{(m)}(t)$ be the power series $V_m(t, x_j^{(m)}(t))$. By construction, we have

$$W_m(t, x_j^{(m)}(t)) \cdot y_j^{(m)}(t) \equiv 1 \mod t^{n+1}$$

and

$$y_j^{(m)}(t) \equiv y_j(t) \mod t^{m+1},$$

for all $j \in \{1, \dots, g\}$.

Proposition 25. *The product $H(X_m(t)) \cdot X_m'(t)$ satisfy the following relation:*

$$H(X_m(t)) \cdot X_m'(t) \mod t^{n+1} = \begin{pmatrix} r_1^{(m)} & r_2^{(m)} & \cdots & r_g^{(m)} \\ r_2^{(m)} & r_3^{(m)} & & r_{g+1}^{(m)} \\ \vdots & & & \\ r_g^{(m)} & r_{g+1}^{(m)} & \cdots & r_{2g-1}^{(m)} \end{pmatrix} \begin{pmatrix} w_0(t) \\ w_1(t) \\ \vdots \\ w_{g-1}(t) \end{pmatrix}$$

Proof. This is a direct consequence of the fact that $1/y_j^{(m)}(t) \mod t^{n+1} = \sum_{i=0}^{g-1} w_i^{(m)}(t) x_j^{(m)}(t)^i$, for all $j \in \{1, \dots, g\}$. □

Corollary 26. *Step 1 can be carried out using at most $\tilde{O}(ng)$ operations in K_0.*

Proof. Approximations of the Newton sums $s_i^{(m)}(t)$ modulo t^{n+1} can be computed from the polynomial $U_m(t, z)$ and using Algorithm 2. By Proposition 22, this can be done using at most $\tilde{O}(ng)$ operations in K_0. The power series $r_i^{(m)}(t)$ can be computed using Equation (18) and the polynomial $W_m(t, z)$ is constructed from the classical Newton scheme for extracting square roots. By Proposition 25, the product $H(X_m(t)) \cdot X_m'(t) \mod t^{n+1}$ is a Hankel matrix-vector product that can be computed using Algorithm 3. By Proposition 24, this can be done using at most $\tilde{O}(ng)$ operations in K_0. □

We now explain how to compute U_n from the linear system $H(X_m(t)) \cdot (X_n(t) - X_m(t)) = F_m(t)$ modulo t^{n+1}. The following proposition and its proof give a construction over $K_0[\![t]\!]$ of an approximation of the interpolating polynomial of the data $\{(x_1^{(m)}(t), x_1^{(n)}(t) - x_1^{(m)}(t)), \dots, (x_g^{(m)}(t), x_g^{(n)}(t) - x_g^{(m)}(t))\}$ modulo t^{n+1}.

Proposition 27. *Let $m \geq 0$ be an integer and $n = 2m + 1$. There exists a polynomial $h_m(t, z) \in \mathcal{O}_K[\![t]\!][z]$ of degree $g - 1$ such that $x_j^{(n)}(t) = h_m(t, x_j^{(m)}(t)) + x_j^{(m)}(t) \mod t^{n+1}$ for all $j \in \{1, \dots, g\}$.*

Proof. We give a construction of h_m using the approach in [28, Section 5]. Let $D_m(t, z)$ be the polynomial defined by

$$D_m(t, z) = x_1^{(m)}(t) z^g + x_2^{(m)}(t) z^{g-1} + \cdots + x_{g-1}^{(m)}(t) z^2 + x_g^{(m)}(t) z.$$

Write

$$U_m(t,z)\cdot D_m(t,z) = q_{2g}^{(m)}(t)z^{2g} + q_{2g-1}^{(m)}(t)z^{2g-1} + \cdots + q_1^{(m)}(t)z + q_0^{(m)}(t)$$

and

$$Q_m(t,z) = q_{2g}^{(m)}(t)z^{g-1} + q_{2g-1}^{(m)}(t)z^{g-2} + \cdots + q_{g+2}^{(m)}(t)z + q_{g+1}^{(m)}(t).$$

For all $j = 1, \cdots, g$ we have the following relation

$$x_j^{(n)}(t) - x_j^{(m)}(t) = \frac{Q_m(t, x_j^{(m)}(t))}{\partial_z U_m(t, x_j^{(m)}(t))}\cdot y_j^{(m)}(t),$$

where $\partial_z U_m$ denotes the partial derivative of U_m with respect to variable z. Hence, we take h_m to be equal to

$$h_m(t,z) = \frac{Q_m(t,z)}{\partial_z U_m(t,z)}\cdot V_m(t,z) \mod (t^{n+1}, U_m(t,z)).$$

□

We end up with the computation of U_n using a Newton scheme. Write $U_n = U_m + T_m$, where $T_m \in t^{m+1}K_0[\![t]\!][z]$. By Propostion 27, the polynomial $U_n(t,z)$ can be constructed from the following relations

$$U_n(t, h_m(t, x_j^{(m)}(t)) + x_j^{(m)}(t)) \mod t^{n+1} = 0, \text{ for all } j \in \{1,\dots,g\}. \tag{19}$$

Equation (19) can be rewritten as follows:

$$U_n(t, x_j^{(m)}(t)) + h_m(t, x_j^{(m)}(t))\cdot \partial_z U_n(t, x_j^{(m)}) \equiv 0 \mod t^{n+1}, \text{ for all } j \in \{1,\dots,g\}. \tag{20}$$

Using the fact that $U_m(t, x_j^{(m)}(t)) = 0$ for all $j \in \{1,\dots,g\}$, we get

$$T_m(t, x_j^{(m)}(t)) + h_m(t, x_j^{(m)}(t))\cdot \partial_z U_m(t, x_j^{(m)}) \equiv 0 \mod t^{n+1}, \text{ for all } j \in \{1,\dots,g\}. \tag{21}$$

Repeating the above calculations in the reverse direction, we obtain the next proposition.

Proposition 28. *Let $m \geq 0$ be an integer and $n = 2m+1$. Let $T_m \in t^{m+1}K_0[\![t]\!][z]$ be the polynomial defined by*

$$T_m(t,z) = h_m(t,z)\cdot \partial_z U_m(t,z) \mod (t^{n+1}, U_m).$$

Then, $U_n = U_m + T_m$ is an approximation of $U(t,z)$ modulo t^{n+1}.

Corollary 29. *Step 3 can be performed using at most $\tilde{O}(ng)$ operations in K_0.*

Proof. This is a direct consequence of Propositions 27 and 28. □

We summarize all the steps that we performed to solve Equation (16) in Algorithm 4.

Algorithm 4: Alternant Differential System Solver

`AlternantSystem` (G, f_2, U_0, V_0, n)

Input : $G \mod t^n$, $f_2 \mod t^n$, $U_0(z) = \prod_{j=1}^{g} (z - x_j^{(0)})$ a separable polynomial, $V_0(z)$ such that $V_0(x_j^{(0)}) = y_j^{(0)}$ for all $j = 1, \dots g$.

Output: The polynomial $U \mod t^{n+1}$ whose roots form the solution of Equation (16)

if $n = 0$ **then**
  return $U_0 \mod t$, $1/V_0 \mod (t, U_0)$

$m := \lceil \frac{n-1}{2} \rceil$;

$X_m, W_m :=$ `AlternantSystem`(G, f_2, U_0, V_0, m);

$W_m := (W_m/2) \cdot (-f_2 \cdot W_m^2 + 3) \mod (t^{n+1}, U_m)$;

$V_m := f_2 \cdot W_m \mod (t^{n+1}, U_m)$;

$s_1^{(m)}, \dots, s_{2g-1}^{(m)} :=$ `NewtonSums`$(U_m, 2g-1, n)$;

for $j := 1$ *to* $2g - 1$ **do**
  $r_i^{(m)} := (d s_i^{(m)}/dt)/i$;

$Hp_1^{(m)}, \dots, Hp_g^{(m)} :=$ `HankelProd`$((r_{i+j-1}^{(m)})_{i,j}, (w_0^{(m)}, \dots, w_{g-1}^{(m)}), n)$;

for $i := 1$ *to* g **do**
  $F_i^{(m)} = \int (G_i - Hp_i^{(m)})\, dt \mod t^{n+1}$;

$D_m := F_1^{(m)} z^g + F_2^{(m)} z^{g-1} + \cdots + F_{g-1}^{(m)} z^2 + F_g^{(m)} z \mod t^{n+1}$;

Write $U_m \cdot D_m = q_{2g}^{(m)} z^{2g} + q_{2g-1}^{(m)} z^{2g-1} + \cdots + q_1^{(m)} z + q_0^{(m)} \mod t^{n+1}$;

$Q_m := q_{2g}^{(m)} z^{g-1} + q_{2g-1}^{(m)} z^{g-2} + \cdots + q_{g+2}^{(m)} z + q_{g+1}^{(m)} \mod t^{n+1}$;

$T_m := -Q_m \cdot V_m \mod (t^{n+1}, U_m)$;

return $U_m + T_m \mod t^{n+1}, W_m \mod t^{n+1}$

Theorem 30. *Let K_0 be an unramified extension of $\mathbb{Q}_p$. Let $n, g \in \mathbb{N}$, $N \in \mathbb{N}^*$, $G \in \mathcal{O}_{K_0}[\![t]\!]^g$, $f_2 \in \mathcal{O}_{K_0}[\![t]\!][z]$ of degree $O(g)$, $U_0 = \prod_{j=1}^{g} (z - x_j^{(0)}) \in \mathcal{O}_{K_0}[z]$, $V_0 \in \mathcal{O}_{K_0}[z]$ of degree $g - 1$ such that $V_0(x_j^{(0)}) \mod p \neq 0$ and U_0 divides $f_2 - V_0^2$ in $\mathcal{O}_{K_0}[\![t]\!][z]$. We assume that U_0 is separable over the the residue field of K_0 and that the polynomial $U(t, z)$, whose roots form a solution of Equation (16), have coefficients in $\mathcal{O}_{K_0}[\![t]\!]$. Then, the procedure `AlternantSystem` runs with fixed point arithmetic at precision $O(p^{M + \lfloor \log_p(2g-1) \rfloor})$, with $M = \max(N, 2) + \lfloor \log_p(n) \rfloor$ if $p = 3$ and $M = N + \lfloor \log_p(n) \rfloor$ otherwise. All the computations are done in $\mathcal{O}_{K_0}$ and the result is correct at precision $O(p^N)$.*

Proof. The proof is similar to the proof of Theorem 3. □

Proposition 31. *When performed with fixed point arithmetic at precision $O(p^M)$, the bit complexity of Algorithm 4 is $\tilde{O}\left(ng \cdot \mathrm{A}(K_0; M)\right)$, where $\mathrm{A}(K_0; M)$ denotes an upper bound on the bit complexity of the arithmetic operations in $\mathcal{O}_{K_0}/p^M\mathcal{O}_{K_0}$.*

Proof. Let D denote the algebraic complexity of Algorithm 4, then we have the following relation

$$\mathrm{D}(n) \leq \mathrm{D}\left(\left\lceil \frac{n-1}{2} \right\rceil\right) + \tilde{O}(ng).$$

Solving the recurrence, we find $\mathrm{D}(n) = \tilde{O}(ng)$. Therefore, the bit complexity of Algoritmh 4 is $\tilde{O}\left(ng \cdot \mathrm{A}(K_0; M)\right)$. □

5. Fast computation of the multiplication by-ℓ maps

Thanks to Algorithm 4, we now have fast algorithms for computing rational representations of separable isogenies between Jacobians of hyperelliptic curves defined over fields of odd characteristic, after having possibly lifted the two curves and the normalization matrix to the p-adics. In this section, we are interested in the computation of a rational representation of the multiplication by an integer. It is therefore necessary to know in advance some bounds on the degrees of its components.

It has been proved that the degrees of the components of a rational representation of the multiplication-by-ℓ are bounded by $O(\ell^2)$, only for curves of genus 2 and 3 [21, Chapter 4]. In the general case, it has been shown that these degrees are bounded by $O_g(\ell^3)$[2] [21, Theorem 4.13], although experiments show that they are only quadratic in ℓ.

In Section 5.1, we use the results of Section 3.2 to reduce the bound to $O(g\ell^2)$. Consequently, we derive from Algorithm 4 a quasi-optimal algorithm to compute a rational representation of the multiplication by an integer.

5.1. Cantor ℓ-division polynomials

Let $C : y^2 = f(x)$ be a hyperelliptic curve of genus g over a finite field k and $\ell > g$ an integer coprime to the characteristic of k.
Let $P \in C(k)$. For a generic point $Q = (x, y)$ on C, the Mumford representation of the element $\ell[Q - P]$ in the Jacobian of C can be written as follows

$$\ell[Q - P] = \left(X^g + \sum_{i=1}^{g-1} \frac{d_i(x)}{d_g(x)} X^i,\ y \sum_{i=1}^{g-1} \frac{e_i(x)}{e_g(x)} X^i \right),$$

where the numerators $d_0, \ldots, d_{g-1}, e_0, \ldots, e_{g-1}$ are polynomials in $k[x]$ and the denominators d_g and e_g are monic polynomials in $k[x]$. Therefore, $\left(\frac{d_0}{d_g}, \ldots, \frac{d_{g-1}}{d_g}, \frac{e_0}{e_g}, \ldots, \frac{e_{g-1}}{e_g} \right)$ is a rational representation of the multiplication-by-ℓ map.

Definition 32. *The $2g + 2$ polynomials $d_0, \ldots, d_g, e_0, \ldots e_g$ are called Cantor's ℓ-division polynomials.*

Since the multiplication-by-ℓ endomorphism is a separable $(\ell^2, \ldots, \ell^2)$-isogeny, we can then apply Propositions 16 and 19 and Remark 3 in order to obtain bounds on the degrees of the Cantor's ℓ-division polynomials. This gives the following result.

Proposition 33. *The degrees of the polynomials $d_0, \ldots, d_g$ are bounded by $g\ell^2$. Moreover,*

[2] the notation O_g means that we are hiding the terms that depend on g

- *if P is a Weierstrass point, then the degrees of $e_0, \dots, e_g$ are bounded by*

$$\begin{cases} \frac{3}{2}g\ell^2 + g + 1 & \text{if } \deg(f) = 2g+1 \\ 2g\ell^2 + g + 1 & \text{otherwise} \end{cases}$$

- *if P is not a Weierstrass point, then the degrees of $e_0, \dots, e_g$ are bounded by*

$$\begin{cases} 3g\ell^2 + g + 1 & \text{if } \deg(f) = 2g+1 \\ 4g\ell^2 + g + 1 & \text{otherwise} \end{cases}$$

Remark 5. The bounds obtained in Propostion 33 are not optimal. In fact, the experiments carried out by Abelard in his thesis [21, Section 4.2] show that Cantor ℓ-division polynomials have degrees slightly smaller than the bounds that we have obtained in Propostion 33.

The next theorem and its proof give an efficient algorithm to compute Cantor's division polynomials.

Theorem 34. *Let p an odd prime number and $g > 1$ an integer. Let ℓ be an integer greater than g and coprime to p. Let $C : y^2 = f(x)$ be a hyperelliptic curve of genus g defined over a finite field k of odd characteristic p. There exists an algorithm that computes Cantor ℓ-division polynomials of C, performing at most $\tilde{O}(\ell^2 g^2)$ operations in k.*

Proof. Let p be the characteristic of k and $d = [k : \mathbb{F}_p]$. For the sake of simplicity, we will assume that C admits a Weierstrass point over k. The algorithm performs the following steps:

1. Pick a Weierstrass point $P \in C(k)$.
2. Chose a point $Q \in C(k)$ different from P such that $\ell[Q - P]$ is generic.
3. Lift C arbitrarily as $\widetilde{C} : y^2 = \tilde{f}(x)$ over an unramified extension K_0 of $\mathbb{Q}_p$ of degree d with a p-adic precision equal to $1 + \lfloor \log_p(2g\ell^2) \rfloor + \lfloor \log_p(2g-1) \rfloor$.
4. Lift P as $\tilde{P}$ and Q as $\tilde{Q}$ over K_0 such that $\tilde{P}, \tilde{Q} \in \tilde{C}(K_0)$.
5. Solve the differential equation (16) by applying Algorithm 4 to the following input:
   - $n = 2g\ell^2$,
   - $f_2 = \tilde{f}$,
   - $U_0(z)$: the first Mumford coordinate of $\ell[\tilde{Q} - \tilde{P}]$,
   - $V_0(z)$: the second Mumford coordinate of $\ell[\tilde{Q} - \tilde{P}]$,
   - $G(t)$, given by the following relation

$$G = \frac{\ell}{v(t)} \begin{pmatrix} 1 \\ u(t) \\ u(t)^2 \\ \vdots \\ u(t)^{g-1} \end{pmatrix},$$

   where $u(t) = t + x_{\tilde{Q}}$ and $v(t) = \sqrt{f(u(t))}$ such that $v(0) = y_{\tilde{Q}}$.

Let $U(t, z)$ be the reduction of the output of the algorithm in k.

6. Reconstruct from $U(t, z)$ the $g + 1$ polynomials $d_0, \dots, d_g$.

7. Recover the polynomials $e_0, \dots, e_g$ from $d_0, \dots, d_g$ and the equation of C.

The time complexity of the algorithm depends mainly on the complexity of steps 5,6 and 7. According to Proposition 31, step 5 can be carried out for a cost of $\tilde{O}(g^2\ell^2)$ operations in k. The $g + 1$ polynomials $d_0, \dots, d_g$ are obtained by reconstructing (for example) d_0/d_g using Padé approximants from the constant coefficient of U then multiplying the other coefficients of U by d_g to recover $d_1, \dots, d_{g-1}$. Therefore, step 6 requires $\tilde{O}(g^2\ell^2)$ operations in k as well. Step 7 is executed as follows: we make use of the polynomials $d_0, \dots, d_g$ to increase the t-adic approximation of the polynomial $U(t, z)$ to $2\deg(e_0)$. We compute, using a Newton iteration, the degree g polynomial $V(t, z)$ such that $V(0, z) = V_0(z)$ and

$$V(z, t)^2 \equiv f(z) \mod (t^{2\deg(e_0)}, U(t, z)).$$

We reconstruct (for example) the rational fraction e_0/e_g from the constant coefficient of V. The polynomials $e_1, \dots, e_{g-1}$ are obtained by multiplying e_g with the non-constant coefficients of V. This can also be carried out using $\tilde{O}(g^2\ell^2)$ operations in k. □

5.2. Experiments

We made an implementation of both Algorithm 4 and the Padé approximant step using the HALF-GCD algorithm given in [29] with the MAGMA computer algebra system [30] to compute Cantor ℓ-division polynomials in $\mathbb{F}_5$ for hyperelliptic curves. Our implementation is available at [31] . Timings are detailed in Figures 1 and 2. All the calculations were done in the ring $\mathbb{Z}_5$ with a fixed precision which is equal to $1 + \lfloor \log_5(2g\ell^2) \rfloor + \lfloor \log_5(2g - 1) \rfloor$. The observed timings fit rather well with the expected time complexity, which is $\tilde{O}(\ell^2 g^2)$: Figure 2 (resp. Figure 1) shows that the time complexity of our algorithm is almost linear in g^2 (resp. in ℓ^2).

References

[1] R. Schoof, Elliptic curves over finite fields and the computation of square roots mod p, Math. Comp. 44 (170) (1985) 483–494. doi:10.2307/2007968.
URL https://doi.org/10.2307/2007968

[2] A. O. L. Atkin, The number of points on an elliptic curve modulo a prime, manuscript, Chicago IL (1988).

[3] F. Morain, Calcul du nombre de points sur une courbe elliptique dans un corps fini: aspects algorithmiques, Journal de Théorie des Nombres de Bordeaux 7 (1) (1995) 255–282.
URL http://www.jstor.org/stable/43972443

[4] R. Schoof, Counting points on elliptic curves over finite fields, Journal de Théorie des Nombres de Bordeaux 7 (1) (1995) 219–254.
URL www.numdam.org/item/JTNB_1995__7_1_219_0/

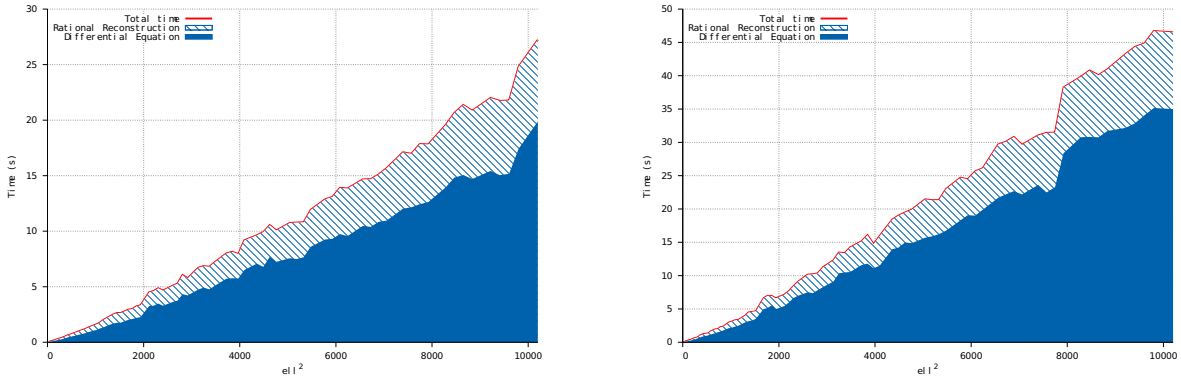
[5] J. Pila, Frobenius maps of abelian varieties and finding roots of unity in finite fields, Mathematics of Computation 55 (1990) 745–763.

[6] P. Gaudry, R. Harley, Counting points on hyperelliptic curves over finite fields, in: W. Bosma (Ed.), Algorithmic Number Theory, Springer Berlin Heidelberg, Berlin, Heidelberg, 2000, pp. 313–332.

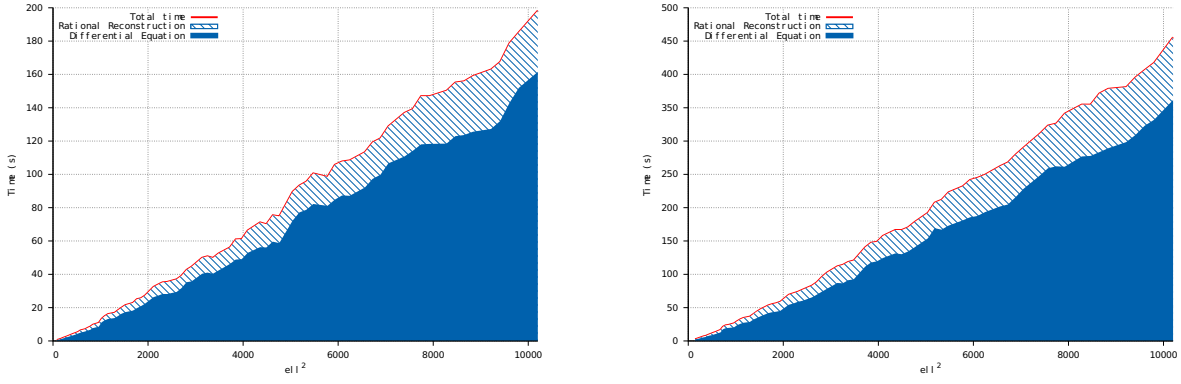
[7] P. Gaudry, É. Schost, Construction of secure random curves of genus 2 over prime fields, in: C. Cachin, J. L. Camenisch (Eds.), Advances in Cryptology - EUROCRYPT 2004, Springer Berlin Heidelberg, Berlin, Heidelberg, 2004, pp. 239–256.

[8] S. Abelard, Counting points on hyperelliptic curves with explicit real multiplication in arbitrary genus, Journal of Complexitydoi:10.1016/j.jco.2019.101440.
URL https://hal.inria.fr/hal-01905580

[9] A. Bostan, F. Morain, B. Salvy, E. Schost, Fast algorithms for computing isogenies between elliptic curves, Math. Comp. 77 (263) (2008) 1755–1778. doi:10.1090/S0025-5718-08-02066-8.
URL https://doi.org/10.1090/S0025-5718-08-02066-8



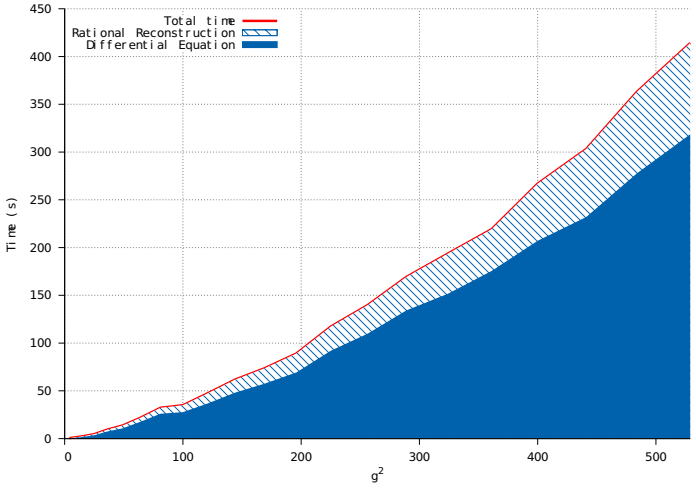
(a) $g = 4$ (b) $g = 5$



(c) $g = 8$ (d) $g = 11$

Timings obtained with MAGMA V2.25-7 on a laptop with an INTEL processor E5-2687WV4@3.00GHZ

Figure 1: Computation of the multiplication-by-ℓ map over $\mathbb{F}_5$ for $\ell \in \{g+1, \ldots, 101\}$ and such that $\gcd(\ell, 5) = 1$



Timings obtained with MAGMA V2.25-7 on a laptop with an INTEL processor E5-2687WV4@3.00GHZ

Figure 2: Computation of the multiplication-by-31 map over $\mathbb{F}_5$ for $g \in \{2, \ldots, 30\}$.

[10] P. Lairez, T. Vaccon, On p-adic differential equations with separation of variables, in: Proceedings of the 2016 ACM International Symposium on Symbolic and Algebraic Computation, ACM, New York, 2016, pp. 319–323.

[11] J.-M. Couveignes, T. Ezome, Computing functions on jacobians and their quotients, LMS Journal of Computation and Mathematics 18 (1) (2015) 555–577. doi:10.1112/S1461157015000169.

[12] X. Caruso, E. Eid, R. Lercier, Fast computation of elliptic curve isogenies in characteristic two, working paper or preprint (Mar. 2020).
URL https://hal.archives-ouvertes.fr/hal-02508825

[13] E. Eid, Sur le calcul d'isogénies par résolution d'équations différentielles p-adiques, Ph.D. thesis, thèse de doctorat dirigée par Lercier, Reynald et Caruso, Xavier Mathématiques et leurs interactions Rennes 1 2021 (2021).
URL http://www.theses.fr/2021REN1S012

[14] J. Kieffer, A. Page, D. Robert, Computing isogenies from modular equations between Jacobians of genus 2 curves, working paper or preprint (Jan. 2020).
URL https://hal.archives-ouvertes.fr/hal-02436133

[15] R. Lercier, T. Sirvent, On Elkies subgroups of l-torsion points in elliptic curves defined over a finite field, J. Théor. Nombres Bordeaux 20 (3) (2008) 783–797.
URL http://jtnb.cedram.org/item?id=JTNB_2008__20_3_783_0

[16] F. OORT, T. SEKIGUCHI, The canonical lifting of an ordinary jacobian variety need not be a jacobian variety, J. Math. Soc. Japan 38 (3) (1986) 427–437. doi:10.2969/jmsj/03830427.
URL https://doi.org/10.2969/jmsj/03830427

[17] X. Caruso, Computations with p-adic numbers, Les cours du CIRM 5 (1). doi:10.5802/ccirm.25.
URL https://ccirm.centre-mersenne.org/item/CCIRM_2017__5_1_A2_0

[18] T. Vaccon, Précision p-adique: applications en calcul formel, théorie des nombres et cryptographie, Ph.D. thesis, University of Rennes 1 (2015).

[19] K. S. Kedlaya, C. Umans, Fast polynomial factorization and modular composition, SIAM J. Comput. 40 (6) (2011) 1767–1802. doi:10.1137/08073408X.
URL https://doi.org/10.1137/08073408X

[20] D. G. Cantor, On the analogue of the division polynomials for hyperelliptic curves. 1994 (447) (1994) 91–146. doi:doi:10.1515/crll.1994.447.91.
URL https://doi.org/10.1515/crll.1994.447.91

[21] S. Abelard, Comptage de points de courbes hyperelliptiques en grande caractéristique : algorithmes et complexité, Ph.D. thesis, thèse de doctorat dirigée par Gaudry, Pierrick et Spaenlehauer, Pierre-Jean Informatique Université de Lorraine 2018 (2018).
URL http://www.theses.fr/2018LORR0104

[22] A. Bostan, F. Chyzak, M. Giusti, R. Lebreton, G. Lecerf, B. Salvy, É. Schost, Algorithmes efficaces en calcul formel, 2017.

[23] X. Caruso, D. Roe, T. Vaccon, Tracking p-adic precision, LMS J. Comput. Math. 17 (suppl. A) (2014) 274–294. doi:10.1112/S1461157014000357.
URL https://doi.org/10.1112/S1461157014000357

[24] X. Caruso, D. Roe, T. Vaccon, p-adic stability in linear algebra, in: ISSAC'15—Proceedings of the 2015 ACM International Symposium on Symbolic and Algebraic Computation, ACM, New York, 2015, pp. 101–108.

[25] J. S. Milne, Abelian varieties, in: Arithmetic geometry, Springer, 1986, pp. 103–150.

[26] T. Matsusaka, On a characterization of a Jacobian variety, 1959.

[27] A. Bostan, L. González-Vega, H. Perdry, É. Schost, From Newton sums to coefficients: complexity issues in characteristic p, in: MEGA'05, 2005, eighth International Symposium on Effective Methods in Algebraic Geometry, Porto Conte, Alghero, Sardinia (Italy), May 27th – June 1st.

[28] E. Kaltofen, L. Yagati, Improved sparse multivariate polynomial interpolation algorithms, in: Symbolic and algebraic computation (Rome, 1988), Vol. 358 of Lecture Notes in Comput. Sci., Springer, Berlin, 1989, pp. 467–474. doi:10.1007/3-540-51084-2_44.
URL https://doi.org/10.1007/3-540-51084-2_44

[29] E. Thomé, Algorithmes de calcul de logarithmes discrets dans les corps finis, Ph.D. thesis, École polytechnique (2003).

[30] W. Bosma, J. Cannon, C. Playoust, The Magma algebra system. I. The user language, J. Symbolic Comput. 24 (3-4) (1997) 235–265, computational algebra and number theory (London, 1993). doi:10.1006/jsco.1996.0125.
URL http://dx.doi.org/10.1006/jsco.1996.0125

[31] E. Eid, Package `Equadif_Solver`, https://github.com/eeid95/HyperellipticIsogeny (2022).