A Point Counting Algorithm for Cyclic Covers of the Projective Line

Cécile Gonçalves

A Point Counting Algorithm for Cyclic Covers of the Projective Line

Cécile Gonçalves

Abstract. We present a Kedlaya-style point counting algorithm for cyclic covers $y^r = f(x)$ over a finite field $\mathbb{F}_{p^n}$ with p not dividing r, and r and $\deg f$ not necessarily coprime. This algorithm generalizes the Gaudry–Gürel algorithm for superelliptic curves to a more general class of curves, and has essentially the same complexity. Our practical improvements include a simplified algorithm exploiting the automorphism of $\mathcal{C}$, refined bounds on the p-adic precision, and an alternative pseudo-basis for the Monsky–Washnitzer cohomology which leads to an integral matrix when $p \geq 2r$. Each of these improvements can also be applied to the original Gaudry–Gürel algorithm. We include some experimental results, applying our algorithm to compute Weil polynomials of some large genus cyclic covers.

1. Introduction

A cyclic cover of the projective line is a nonsingular projective curve $\mathcal{C}$ defined over $\mathbb{F}_{p^n}$ by the affine plane model

$$\mathcal{C} : y^r = f(x),$$

where f is a monic, squarefree degree d polynomial and p does not divide r. Counting points on, and more generally determining the zeta function of a cyclic cover is an interesting problem with many applications in number theory.

A lot of work has been done in point counting during the last three decades, providing efficient point counting algorithms for elliptic curves ($r = 2$ and $d = 3$). The first deterministic polynomial time algorithm for counting points of elliptic curves was the ℓ-adic algorithm of Schoof [**Sch85**]. It was improved by Atkin and Elkies to give the famous SEA algorithm (see [**Sch95**] for the details). Pila proposed a generalization of this approach to general abelian varieties [**Pil90, Pil88**], and Schoof-style algorithms have been implemented with success for hyperelliptic curves of genus 2 ($r = 2, d \leq 6$) [**GS12, GKS11**], but there seems to be no hope of a practical ℓ-adic point counting algorithm for $d > 6$ or $r > 2$, since these algorithms are exponential in the genus. Other efficient point counting algorithms using canonical lifts or the AGM also seem limited to genus 1 and 2, and they are also exponential in $\log p$ [**Sat00, GH00, Mes00, Mes02**].

In 2001, Kedlaya [**Ked01**] published a point counting algorithm for odd degree hyperelliptic curves ($r = 2, d = 2g + 1$) over finite fields of small characteristic p. This algorithm uses a lift of the Frobenius on Monsky–Washnitzer cohomology

2010 *Mathematics Subject Classification.* Primary 14G05, 11G20; Secondary 14G15, 68Q25, 14G10, 11Y16, 14Q15.

Key words and phrases. Algebraic geometry, Number Theory.

The author was supported in part by DGA (Délégation Générale de l'Armement), France.

(see [**vdP86**] for details) and is polynomial in the genus and the field degree, but linear in p (and hence exponential in $\log p$). This complexity was improved by Harvey [**Har07**] in larger characteristic reducing the dependence to $\sqrt{p}$. Kedlaya's algorithm has been extended to more general classes of curves including superelliptic curves [**GG01**], $\mathcal{C}_{a,b}$ curves [**DV06**] and nondegenerate curves [**CDV06**]. In the latter two cases, the algorithm is slower than for superelliptic curves of the same genus, because the curves involved are too much general to find a convenient basis of cohomology. This problem can be solved using deformation theory [**CHV08**] which reduces the computation of the Weil polynomial of a $\mathcal{C}_{a,b}$ curve into the computation of the Weil polynomial of a superelliptic curve. This algorithm has been extended to hypersurfaces [**PT13**]. Minzlaff [**Min10**] improved the complexity of the Gaudry–Gürel algorithm for superelliptic curves applying the improvements of Harvey. Tuitman [**Tui14**] has recently proposed a Kedlaya-style algorithm for general covers of the projective line. An alternate approach using Serre duality [**BEd13**] is available for smooth curves over finite fields and appears promising for the general case.

In this paper, we present a Kedlaya–style algorithm for cyclic covers of the projective line which runs in

$$\widetilde{O}\left(pn^3d^4r^3+n^2rd^{\nu+1}\left(\sum_{i=1}^{s}c_i^{\nu}\right)\right)$$

elementary operations, where the permutation $j\mapsto p^nj \bmod r$ of $\{1,\cdots,r-1\}$ is a product of s cycles of lengths $c_1,c_2,\cdots,c_s$; ν is the exponent in the complexity of matrix multiplication ($2<\nu<3$), and $\widetilde{O}$ is the Soft-Oh notation which ignores the logarithmic factors. Note that in the best case, which is precisely the case where the r-th roots of unity are contained in $\mathbb{F}_{p^n}$, we have $s=r-1$ and $c_i=1$ for $1\leq i\leq s$, which leads to a complexity in $\widetilde{O}\left(pn^3d^4r^3\right)$ elementary operations. In the worst case, $s=1$ and $c_1=r-1$, which leads to a complexity in $\widetilde{O}\left(pn^3d^4r^3+n^2r^{\nu+1}d^{\nu+1}\right)$ elementary operations. Note also that this approach is compatible with Harvey's improvements so we can reduce the dependency in p to $\sqrt{p}$ in larger characteristic.

This algorithm generalizes the Gaudry–Gürel algorithm [**GG01**] from superelliptic curves to general cyclic covers, following the approach of Harrison [**Har12**] for even-degree hyperelliptic curves. While our algorithm has essentially the same complexity as the Gaudry–Gürel algorithm, we offer practical improvements including

- a simplified algorithm exploiting the automorphism of $\mathcal{C}$;
- refined bounds on the p-adic precision; and
- an alternative pseudo-basis for the Monsky–Washnitzer cohomology which leads to an integral matrix when $p\geq 2r$.

Each of these improvements can also be applied to the original Gaudry–Gürel algorithm.

The paper is organized as follows: Section 2 recalls the definition of cyclic covers with their main properties, Section 3 describes Monsky–Washnitzer cohomology for cyclic covers and the action of Frobenius. Section 4 gives a summary of our algorithm; in Section 5 we analyze the complexity of our algorithm. Section 6 proves the precision bounds to which we have to perform the computations in order to have an exact result (these bounds also apply to the Gaudry–Gürel algorithm); in Section 7 we study the use of another pseudo-basis which leads to a matrix with integral coefficients when $p\geq 2r$; the use of this basis slightly accelerates the computations. To conclude, Section 8 gives some numerical experiments.

2. Cyclic covers of the projective line

DEFINITION 2.1. A cyclic cover of the projective line is a nonsingular projective curve $\mathcal{C}$ defined over $\mathbb{F}_q$ by the affine plane model

$$\mathcal{C} : y^r = f(x), \tag{2.1}$$

where f is a monic, squarefree degree d polynomial over $\mathbb{F}_q$ and the characteristic p of $\mathbb{F}_q$ does not divide r.

Let

$$\delta := \gcd(r, d).$$

A cyclic cover $\mathcal{C} : y^r = f(x)$ embeds naturally in the weighted projective space $\mathbb{P}\left(\frac{r}{\delta}, \frac{d}{\delta}, 1\right)$ (see [**Rei02**] for details on weighted projective spaces), where it is a nonsingular curve with δ points at infinity. The genus of the curve is

$$g = \frac{(r-1)(d-1)}{2} - \frac{\delta - 1}{2}. \tag{2.2}$$

It is also equipped with an automorphism of order r defined by

$$\rho_r : (x, y) \longmapsto (x, \zeta_r y)$$

where ζ_r is a primitive r-th root of unity in $\overline{\mathbb{F}}_q$.

REMARK 2.2. When r and d are coprime, then $\delta = 1$ and $\mathcal{C}$ is superelliptic.

DEFINITION 2.3. Let $\mathcal{C}$ be a genus g curve defined over $\mathbb{F}_q$. The Weil polynomial P of $\mathcal{C}$ is the characteristic polynomial of the q-th power Frobenius acting on the Jacobian $J(\mathcal{C})$ of $\mathcal{C}$ and has the form

$$P(t) = t^{2g} + a_1 t^{2g-1} + \cdots + a_{g-1} t^{g+1} + a_g t^g + q a_{g-1} t^{g-1} + \cdots + q^{g-1} a_1 t + q^g,$$

with $|a_i| \leq \binom{2g}{i} q^{i/2}$. We call the coefficients $(a_1, \ldots, a_g)$ the Weil coefficients of $\mathcal{C}$.

The aim of any point counting algorithm is to compute the Weil polynomial of the given curve. The Weil polynomial determines the cardinality of the Jacobian, since $\#J(\mathcal{C})(\mathbb{F}_q) = P(1)$. Since the Weil polynomial is the reciprocal polynomial of the numerator of the zeta function, it also determines $\#\mathcal{C}(\mathbb{F}_{q^k})$ (and $\#J(\mathcal{C})(F_{q^k})$) for all $k > 0$.

3. Monsky–Washnitzer cohomology for cyclic covers and the action of Frobenius

Let $\mathcal{C} : y^r = f(x)$ be a cyclic cover of the projective line of genus g over $\mathbb{F}_q$, with $q = p^n$ and let d denote the degree of f.

We focus on the case where r and d are not coprime, and f has no root in $\mathbb{F}_q$. This is because if r and d are coprime, then we can simply apply the Gaudry–Gürel algorithm to $\mathcal{C}$. If f has a root α in $\mathbb{F}_q$, then we can immediately reduce to the case where r and d are coprime. Indeed, let f_1 be such that $f(x) = (x - \alpha) f_1(x)$, $h(x) = f_1(x + \alpha)$ and $\mathcal{C}'$ be the superelliptic curve defined over $\mathbb{F}_q$ by

$$\mathcal{C}' : y_1^r = x_1^{d-1} h\left(\frac{1}{x_1}\right).$$

Then the $\mathbb{F}_q$–isomorphism from $\mathcal{C}'$ to $\mathcal{C}$ defined by

$$(x_1, y_1) \longmapsto \left(\frac{1}{x_1 - \alpha}, \frac{y_1}{(x_1 - \alpha)^{d/r}}\right)$$

allows us to apply the Gaudry–Gürel algorithm to $\mathcal{C}'$ to compute $P(t)$ (the algorithm we describe below reduces to Gaudry–Gürel in the superelliptic case). Note that in general, f has no root in $\mathbb{F}_q$ so we can't use this trick.

Recall that $\mathcal{C}$ has δ points at infinity. Their coordinates in $\mathbb{P}(\frac{r}{\delta}, \frac{d}{\delta}, 1)$ are

$$P_{\infty,k} = [1 : \zeta_r^k : 0] \text{ for } 1 \leq k \leq \delta, \tag{3.1}$$

where ζ_r is a primitive r-th root of unity over $\overline{\mathbb{F}}_q$.

In any Kedlaya-style algorithm, we compute in the ring $\mathbb{Z}_q$, which is the ring of integers of $\mathbb{Q}_q$, an unramified extension of $\mathbb{Q}_p$ of degree n. We have

$$\mathbb{Z}_q \cong \mathbb{Z}_p[x]/\langle Q(x)\rangle,$$

where Q is an arbitrary lift to $\mathbb{Z}_p$ of a defining polynomial of $\mathbb{F}_q$ over $\mathbb{F}_p$. The Galois group of $\mathbb{Q}_q$ over $\mathbb{Q}_p$ is cyclic; its generator σ reduces modulo p to the p-th power Frobenius automorphism of $\mathbb{F}_q$.

The aim of our algorithm is to compute the action of Frobenius on the first Monsky–Washnitzer cohomology group of $\mathcal{C}$, denoted by $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$, and defined by

$$H^1_{MW}(\mathcal{C}, \mathbb{Q}_q) = \left((\mathcal{A}^\dagger dx + \mathcal{A}^\dagger dy)/d\mathcal{A}^\dagger\right) \otimes_{\mathbb{Z}_q} \mathbb{Q}_q,$$

where $\mathcal{A}^\dagger = \mathbb{Z}_q^\dagger[[x,y]]/\langle y^r - \overline{f}(x)\rangle$ with $\overline{f}$ an arbitrary lift of degree $d = \deg(f)$ of f to $\mathbb{Z}_q$ and $\mathbb{Z}_q^\dagger[[x,y]]$ is the ring of overconvergent series over $\mathbb{Z}_q$, that is the ring of power series $\sum_{i,j} a_{i,j}x^i y^j$ whose radius of convergence is greater than one. This means that the p-adic valuations of the coefficients grows up at least linearly, that is

$$\exists \alpha, \beta \in \mathbb{R}, \alpha > 0 \text{ such that } \operatorname{ord}_p(a_{i,j}) \geq \alpha|i+j| + \beta, \ \forall i, j.$$

Taking a lowbrow point of view, $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$ consists of differential forms of $\mathcal{C}$ lifted to $\mathbb{Z}_q$ modulo the relations coming from the equation of $\mathcal{C}$ and the fact that $d\phi \equiv 0$ for all rational functions ϕ. See [**vdP86**] for more detailed explanations of $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$.

The first thing to do is to determine a basis of $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$. In order to get a basis which is easily described and convenient to compute with, we remove from $\mathcal{C}$ the ramification points of the projection map on the x-axis $\pi : \mathcal{C} \to \mathbb{P}^1$ defined by $\pi : [x : y : z] \mapsto [x : z]$. Let $\widetilde{\mathcal{C}}$ be the curve corresponding to $\mathcal{C}$ without the $\mathbb{F}_q$-rational divisor formed by the d points on the x-axis and the δ points at infinity (which map to a single rational point of $\mathbb{P}^1$ under π):

$$\widetilde{\mathcal{C}} = \mathcal{C} \ \setminus \ \left(\left\{ (\alpha, 0) \in \mathcal{C}(\overline{\mathbb{F}}_q) \mid f(\alpha) = 0 \right\} \cup \left\{ P_{\infty,k} \mid k \in [1, \delta] \right\} \right).$$

The elements of $H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)$ are the $\sum_{i\in\mathbb{Z}, j\in\mathbb{Z}} (a_{i,j}dx + b_{i,j}dy)x^i y^j$. Using the equation of the curve and the relation

$$ry^{r-1}dy = \overline{f}'(x)dx, \tag{3.2}$$

it follows that the elements of $H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)$ are represented by series in the form $\sum_{0\leq i<d, j\in\mathbb{Z}} a_{i,j}x^i y^j dx$. As Equation (3.2) gives $rx^i y^{j+r-1}dy = x^i y^j \overline{f}'(x)dx$ for all $i, j \in \mathbb{Z}$, and using the fact that $d\left(-\frac{r}{r-j}x^i y^{r-j}\right) = 0$ for $j > r$, we get

$$x^i \overline{f}'(x)\frac{dx}{y^j} = \frac{ri}{j-r}x^{i-1}\frac{dx}{y^{j-r}} \text{ for } j > r. \tag{3.3}$$

If $j \geq 0$, then the relation $y^r = \overline{f}(x)$ implies that the elements of $H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)$ are represented by sums in the form $\sum_{0\leq i<2d, 1\leq j\leq r} a_{i,j}x^i \frac{dx}{y^j}$. Using $d(rx^i y^{r-j}) = 0$, we get

$$\left(rix^{i-1}\overline{f}(x) + (r-j)x^i \overline{f}'(x)\right)\frac{dx}{y^j} = 0 \text{ for } 1 \leq j \leq r \text{ and } i \geq 0. \tag{3.4}$$

As the degree in x of the expression above is $d+i-1$, it follows that

$$\widetilde{B} = \left\{ x^i \frac{dx}{y^j} \mid i \in [0, d-2], j \in [1, r] \right\}$$

is a basis of $H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)$.

Proposition 6.1 of [**vdP86**] shows that working with $\widetilde{\mathcal{C}}$ instead of $\mathcal{C}$ enlarges the dimension of the first Monsky–Washnitzer cohomology group:

$$\dim\left(H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)\right) = 2g + d + \delta - 1 = \dim\left(H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)\right) + d + \delta - 1.$$

The space $H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)$ decomposes into a direct sum of r eigenspaces under the action of the automorphism $\rho_r : (x, y) \mapsto (x, \zeta_r y)$ of $\mathcal{C}$:

- $H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)^1$, which corresponds to the fixed points of ρ_r and has dimension d with basis $\left\{ x^i \frac{dx}{y^r} \mid i \in [0, d-1] \right\}$. This subspace comes from the d points removed from the affine part of $\mathcal{C}$ on the x-axis.
- the $r-1$ spaces $H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)^{\zeta_r^{-j}}$ of dimension $d-1$, for $1 \leq j < r$, with basis
$$\left\{ x^i \frac{dx}{y^j} \mid i \in [0, d-2] \right\}.$$

Observe that $\rho_r\left(x^i \frac{dx}{y^j}\right) = \zeta_r^{-j} x^i \frac{dx}{y^j}$, so ρ_r has eigenvalue ζ_r^{-j} on $H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)^{\zeta_r^{-j}}$ for any $1 \leq j \leq r$.

Let $i : H^1_{MW}(\mathcal{C}, \mathbb{Q}_q) \hookrightarrow H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)$ be the embedding from $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$ to $H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)$. As $i(H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)) \cap H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q) = \{0\}$, the space $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$ is contained in the direct sum of the $H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)^{\zeta_r^j}$, for $1 \leq j < r$ which has dimension $2g + \delta - 1$ with basis

$$B = \left\{ x^i \frac{dx}{y^j} \mid i \in [0, d-2] \, , j \in [1, r-1] \right\}.$$

Clearly B does not correspond to a basis for $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$ (since it contains $\delta - 1$ too many vectors). The space generated by B decomposes into a direct sum of two subspaces stable under the action of Frobenius. Let $c : \langle B \rangle \to H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$ be the map sending an element to its representative in $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$. As $H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)$ contains $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$ and $i(H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)) \cap H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)^1 = \{0\}$ (where i is the embedding from $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$ to $H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)$), it follows that c is surjective. So we have the decomposition

$$\langle B \rangle = H^1_{MW}(\mathcal{C}, \mathbb{Q}_q) \oplus \ker(c).$$

So we will proceed into two steps: we first compute the action of Frobenius on the space generated by B, and then we remove an extra factor from its characteristic polynomial which corresponds to the action of Frobenius on the $(\delta-1)$-dimensional subspace $\ker(c)$. Theorem 4.1 describes this extra factor. This is similar to Harrison's approach in extending Kedlaya's algorithm to hyperelliptic curves with even degree [**Har12**].

Remark 3.1. The p-th power Frobenius maps the space $H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)^{\zeta_r^j}$ to $H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)^{\zeta_r^\ell}$ for $1 \leq j < r$ where $\ell = jp \bmod r$. Thus, the matrix of the p-th power Frobenius map acting on B is a block matrix with exactly $r-1$ non-zero blocks of size $d-1$. There is exactly one non-zero block on each row partition and each column partition.

We begin by computing the action of Frobenius on the elements of B. Let

$$\tau = y^{-r}.$$

We can lift the p-th power Frobenius to differential forms by setting

$$\mathcal{F}(x) = x^p$$

and

$$\mathcal{F}(y)^r = \mathcal{F}(\overline{f}(x))$$

so

$$\begin{aligned} \mathcal{F}(y) &= y^p \left(1 + \left(\mathcal{F}\left(\overline{f}(x)\right) - \overline{f}(x)^p\right)\tau^p\right)^{\frac{1}{r}} \\ &= y^p \sum_{i\geq 0} \binom{1/r}{i} \left(\mathcal{F}\left(\overline{f}(x)\right) - \overline{f}(x)^p\right)^i \tau^{pi}, \end{aligned} \tag{3.5}$$

and

$$\mathcal{F}(dx) = d(\mathcal{F}(x)) = px^{p-1}dx.$$

The p-th power Frobenius acts on the basis vectors as

$$\begin{aligned} \mathcal{F}\left(x^i\frac{dx}{y^j}\right) &= px^{p(i+1)-1}\mathcal{F}(y)^{-j}dx \\ &= px^{p(i+1)-1}y^{-jp}(1+pE(x)\tau^p)^{-j/r}dx \\ &= x^{p(i+1)-1}y^{-jp}\sum_{k\geq 0}\binom{-j/r}{k}p^{k+1}E^k(x)\tau^{pk}dx \\ &= x^{p(i+1)-1}\sum_{k\geq 0}\binom{-j/r}{k}p^{k+1}E^k(x)\tau^{pk+a}\frac{dx}{y^\ell}, \end{aligned} \tag{3.6}$$

where $E(x) = \frac{\mathcal{F}(\overline{f}(x))-\overline{f}(x)^p}{p}$ and $jp = ar+\ell$. Applying a change of index, we obtain

$$\mathcal{F}\left(x^i\frac{dx}{y^j}\right) = x^{p(i+1)-1}\sum_{k\geq a, k\equiv a \bmod p} p^{\frac{k-a}{p}+1}Q_k(x)\tau^k\frac{dx}{y^\ell}.$$

After normalizing (using the equation of $\mathcal{C}$ to remove all the terms with degree in x greater than $\deg(\overline{f})$), and using the fact that $Q_a = 1$, we have:

$$\mathcal{F}\left(x^i\frac{dx}{y^j}\right) = \sum_{k\geq k_0, k\equiv a \bmod p} p^{\frac{k-a}{p}+1}R_k(x)\tau^k\frac{dx}{y^\ell}, \text{ where } k_0 \geq a - \left\lfloor \frac{p(i+1)-1}{d}\right\rfloor \tag{3.7}$$

with $\deg(R_k) < d$ for all $k > 1$.

We then apply two reduction steps described below, resulting from relations in cohomology, in order to express the image of each basis element as a linear combination of the elements of B.

Red1: Decrease the degree in τ by at least one, using the formula

$$R_k(x)\tau^k\frac{dx}{y^\ell} = \left(A_k(x) + \frac{r}{r(k-1)+\ell}B_k'(x)\right)\tau^{k-1}\frac{dx}{y^\ell}, \tag{3.8}$$

where $R_k = A_k\overline{f} + B_k\overline{f}'$, coming from Equation (3.3). We apply (3.8) as many times as needed. Note that A_k and B_k exist because f, and therefore $\overline{f}$ are squarefree.

Red2: Given an expression of the form $S(x)\frac{dx}{y^\ell}$ with S of degree m, use Equation (3.4) to decrease by at least one the degree of S to at most $d-2$ by subtracting some multiples of

$$r(i-d+1)x^{i-d}\overline{f}(x) + (r-\ell)x^{i-d+1}\overline{f}'(x) \text{ for } d-1 \leq i \leq m = \deg(S) \tag{3.9}$$

from S. We apply (3.9) as many times as needed.

We obtain a matrix $M_{\mathcal{F}}$, which is the matrix of the p-th power Frobenius with respect to B. We recover the matrix of the q-th power Frobenius which is

$$M = M_{\mathcal{F}} \cdot M_{\mathcal{F}}^{\sigma} \cdots M_{\mathcal{F}}^{\sigma^{n-1}}, \tag{3.10}$$

where σ is the p-th power Frobenius on $\mathbb{Q}_q$ and $M_{\mathcal{F}}^{\sigma}$ is the matrix obtained by applying σ to the coefficients of $M_{\mathcal{F}}$. Note that we can use the special block structure of $M_{\mathcal{F}}$ to speed up the computation of M, which is a matrix composed of $(d-1) \times (d-1)$ blocks as well.

4. Adaptation of the Gaudry–Gürel algorithm to general cyclic covers

We want to compute the Weil polynomial of $\mathcal{C}$. Once we have computed the characteristic polynomial χ_M, we need to remove an extra factor.

THEOREM 4.1. *The Weil polynomial P of $\mathcal{C}$ is*

$$P(t) = \frac{\chi_M(t)}{U(t)},$$

where $\chi_M(t)$ is the characteristic polynomial of the matrix M corresponding to the action of the q-th power Frobenius with respect to B and

$$U(t) = \prod_{i|\delta\ ,\ i>1} (t^{k_i} - q)^{\frac{\varphi(i)}{k_i}},$$

where k_i is the order of q in $\mathbb{Z}/\varphi(i)\mathbb{Z}$ and φ is the Euler totient function.

PROOF. We follow the approach of [**Har12**, Lemma 3.1].

Let $\alpha_1, \cdots, \alpha_{2g}$ denote the roots of P and $S_e = \alpha_1^e + \alpha_2^e + \cdots + \alpha_{2g}^e$. Let R_e denote the number of roots of f in $\mathbb{F}_{q^e}$ and I_e the number of $\mathbb{F}_{q^e}$-points at infinity. Finally, let

$$L = \pi(\widetilde{\mathcal{C}}),$$

where $\pi : \mathcal{C} \to \mathbb{P}^1$ is the projection on the x-axis.

For every $e > 0$, we have

$$\#\mathcal{C}(\mathbb{F}_{q^e}) = (q^e + 1 - S_e),$$

$$\#\widetilde{\mathcal{C}}(\mathbb{F}_{q^e}) = (q^e + 1 - S_e) - R_e - I_e,$$

and

$$\#L(\mathbb{F}_{q^e}) = (q^e + 1) - R_e - 1.$$

The Lefschetz trace formula for $\widetilde{\mathcal{C}}$ says that for each $e > 0$,

$$\begin{aligned}(q^e + 1 - S_e) - R_e - I_e = &\operatorname{Tr}((q\mathcal{F}^{-1})^e | H^0_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)) \\ &- \operatorname{Tr}((q\mathcal{F}^{-1})^e | H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)^1) \\ &- \sum_{j \neq 0} \operatorname{Tr}((q\mathcal{F}^{-1})^e | H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)^{\zeta_r^{-j}})\end{aligned}$$

while the trace formula for L says that for each $e > 0$,

$$(q^e + 1) - R_e - 1 = \operatorname{Tr}((q\mathcal{F}^{-1})^e | H^0_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)) - \operatorname{Tr}((q\mathcal{F}^{-1})^e | H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)^1).$$

Subtracting the first equation from the second, we get

$$\sum_{j \neq 0} \operatorname{Tr}((q\mathcal{F}^{-1})^e | H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)^{\zeta_r^{-j}}) = S_e + I_e - 1.$$

The sum of the e-th powers of the eigenvalues of $q\mathcal{F}^{-1}$ on $\oplus_j H^1_{MW}(\widetilde{\mathcal{C}}, \mathbb{Q}_q)^{\zeta_r^{-j}}$ is $S_e + I_e - 1$.

Let

$$V(t) = \prod_{i|\delta, i>1} (t^{k_i} - 1)^{\frac{\varphi(i)}{k_i}},$$

where k_i is the order of q in $\mathbb{Z}/\varphi(i)\mathbb{Z}$ and φ is the Euler totient function. Then the e-th power sum of the roots of $V(t)$ is $I_e - 1$ for each $e > 0$, so the e-th power sum of the roots of $V(t)P(t)$ is $S_e + I_e - 1$, or each $e > 0$. Then

$$P(t)V(t) = \chi_{q\mathcal{F}^{-1}}(t),$$

and hence

$$P(t)U(t) = \chi_M(t).$$

□

REMARK 4.2. Note that in the superelliptic case, the fact that $\delta = 1$ tells us that B corresponds to a basis of $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$. So in the superelliptic case we do not have to remove an extra factor from the characteristic polynomial of the Frobenius map acting on B: the characteristic polynomial is the Weil polynomial of $\mathcal{C}$.

In practice, we want to compute the Weil polynomial of $\mathcal{C}$ over $\mathbb{Z}_q$, but p-adic numbers are infinite series so algorithmically we are forced to work with finite approximations. In practice, we work up to a certain precision N: that is, modulo $p^N\mathbb{Z}_q$ for some suitably large value of N.

The Weil bounds (see Definition 2.3) tell us that if we do the computations to sufficient precision then we can recover the Weil polynomial exactly. As we lose some digits of precision during the reduction steps, we have to enlarge these bounds. Theorem 4.3 states the precision bounds and its proof can be found in §6.

THEOREM 4.3. *In order to compute the Weil polynomial of $\mathcal{C}$ exactly, we have to do the computations with the basis B up to precision*

$$N = \min_{n\in\mathbb{N}} \left\{ n - \left\lfloor \log_p\Big(p(rn-1) - r\Big) \right\rfloor \geq N_0 + \left\lfloor \log_p\Big(\frac{dp(r-1)+r}{\delta}\Big) \right\rfloor \right\}$$

where

$$N_0 = \left\lceil \log_p\left(2\binom{2g}{g} q^{g/2}\right) \right\rceil.$$

We will carry out our computations modulo p^N, where N is defined in Theorem 4.3. At the end of the algorithm, we will have determined the Weil polynomial of $\mathcal{C}$ modulo p^{N_0}, which is sufficient to determine it exactly, because of the Weil bounds.

Algorithm 1, CYCLICCOVERWEILPOLYNOMIAL computes the Weil polynomial of a cyclic cover $\mathcal{C}$ defined over $\mathbb{F}_q$ by the equation $y^r = f(x)$. Note that Steps 1 to 5 of Algorithm 1 reduces to Gaudry and Gürel's algorithm.

Step 1 computes the precision N stated by Theorem 4.3. In Step 2, we compute $\mathcal{F}(y)^{-1} \bmod p^N$ using Equation (3.5). Indeed, if $R = 1 + (\mathcal{F}(\overline{f}(x)) - \overline{f}(x)^p)\tau^p$, then $\mathcal{F}(y)^{-1} = y^{-p}R^{-\frac{-1}{r}}$, where $\overline{f}$ is an arbitrary degree $d = \deg(f)$ lift of f to $\mathbb{Z}_q$ and $\tau = y^{-r}$. Note that we use a Newton iteration to compute the inverse of the r-th root of R up to precision N. In Step 3, we compute the action of Frobenius on the vectors of B given by Equation (3.6). We then apply Algorithm 2, REDUCTIONCOHOMOLOGY, to reduce a differential form back to a linear combination of vectors of B using the reduction rules Red1 and Red2 described above in Equations (3.8) and (3.9). The result of Step 3 is the matrix $M_{\mathcal{F}}$ of the p-th power Frobenius map acting on B. The ordering of the indices $\mathcal{J}$ is chosen in such a way that the $d-1$ by $d-1$ blocks are grouped into larger blocks reflecting the cyclic action of multiplication by q modulo r. These larger blocks form a block diagonal matrix: for each cycle of length c, there is a block of size $c(d-1)$. In Step 4, we compute the

matrix M of the q-th power Frobenius map acting on B from $M_{\mathcal{F}}$ using Equation (3.10) and using the block structure of $M_{\mathcal{F}}$. Note that the resulting matrix M has the same block structure. We then compute its characteristic polynomial up to precision N_0. Again, we use the block structure of M. Finally, in Step 5 we compute the extra factor U using Theorem 4.1 and we then return the Weil polynomial of $\mathcal{C}$.

Algorithm 1 CyclicCoverWeilPolynomial

Input: Cyclic cover $\mathcal{C}$ of genus g over $\mathbb{F}_q$, with $q = p^n$, defined by the equation $y^r = f(x)$, with f a monic, squarefree degree d polynomial.
Output: The Weil polynomial $P(t)$ of $\mathcal{C}$.
Step 1: Precision bounds:
Compute $\delta := \gcd(r, d)$;
$N_0 := \left\lceil \log_p \left(2\binom{2g}{g} q^{g/2} \right) \right\rceil$;
$N := \min \left\{ n - \left\lfloor \log_p \left(p(rn-1) - r \right) \right\rfloor \geq N_0 + \left\lfloor \log_p \left(\frac{dp(r-1)+r}{\delta} \right) \right\rfloor \mid n \in \mathbb{N} \right\}$;
Step 2: Compute $\mathcal{F}(y)^{-1} \bmod p^N$:
$R := 1 + (\mathcal{F}(\overline{f}(x)) - \overline{f}(x)^p)\tau^p$; *// where τ is y^{-r} and $\overline{f}$ is an arbitrary lift of f to $\mathbb{Z}_q$.*
$S := R^{-\frac{1}{r}}$; *//$\mathcal{F}(y)^{-1} = y^{-p}S$*
Step 3: Action of Frobenius on B :
//where $B = \left\{ x^i \frac{dx}{y^j} \mid i \in [0, d-2] \, , j \in [1, r-1] \right\}$.
Compute $\mathcal{J}$ as the sequence $[1, r-1]$ sorted in cycles under the action of multiplication by $q \bmod r$. *//We have $J[i+1] = qJ[i] \bmod r$.*
for j in $\mathcal{J}$ **do**
  for $i = 0$ to $d-2$ **do**
    $\omega := px^{p(i+1)-1} S^j \tau^{\frac{jp-\ell}{r}} \frac{dx}{y^\ell}$; *// where $\ell = jp \bmod r$.*
    // $\omega = \mathcal{F}\left(x^i \frac{dx}{y^j} \right)$ has the form $\sum_{0 \leq k \leq \mu} R_k(x) \tau^k \frac{dx}{y^\ell}$, where $\mu = pN - 1$.
    Red:= ReductionCohomology$(\mathcal{C}, N, \omega)$;
    //Then, fill the matrix
    for $k = 0$ to $d-2$ **do**
      $M_{\mathcal{F}}[(d-1)(j-1) + i + 1][(d-1)(\ell-1) + k + 1] := \text{Coeff}(k, \text{Red})$;
    end for
  end for
end for
Step 4: Compute the characteristic polynomial:
$M := M_{\mathcal{F}} . M_{\mathcal{F}}^{\sigma} \ldots M_{\mathcal{F}}^{\sigma^{n-1}}$; *//Use the block structure of $M_{\mathcal{F}}$ to speed up the computation of M*
$\chi(t) := \chi_M(t) \mod N_0$; *//Use the block structure of M to speed up the computation of χ*
Step 5: Remove the extra factor:
$U(t) := \prod_{i \mid \delta \, , \, i > 1} (t^{k_i} - q)^{\frac{\varphi(i)}{k_i}}$, where $k_i := \min \{ n \in N : \varphi(i) \mid q^n - 1 \}$;
return $P(t) := \frac{\chi(t)}{U(t)}$;

5. Complexity

In this section, we describe the space and time complexity of Algorithm 1 as a function of the parameters p, n, r and d and we show that this complexity is linear in p and polynomial in n, r and d (in particular, it is polynomial in the genus). We use the Soft-oh notation so that the logarithmic terms are not taken into account. We let $2 < \nu < 3$ be the exponent such that the complexity of multiplying two square matrices of size k over a ring $\mathcal{R}$ is $\widetilde{O}(k^\nu)$ operations in $\mathcal{R}$

Algorithm 2 ReductionCohomology

Input: A lift of a cyclic cover $\mathcal{C} : y^r = f(x)$ to $\mathbb{Z}_q$ up to precision N, with $q = p^n$, and f a monic, squarefree degree d polynomial, precision N, differential form $\omega = \sum_{0\leq k\leq \mu} R_k(x)\tau^k \frac{dx}{y^\ell}$.

Output: A differential form $T(x)\frac{dx}{y^\ell}$ equivalent to ω, with $\deg(T) \leq d-2$.

Step 1: Reduce degree in τ, that is, transform ω to $S(x)\frac{dx}{y^\ell}$:

// At each iteration, we reduce the degree in τ by at least one.

for $k = \mu$ to 1 **do**

Compute A_k and B_k such that $R_k = A_k\overline{f} + B_k\overline{f}'$, using the extended Euclidean algorithm. *//$\overline{f}$ is an arbitrary lift of f to $\mathbb{Z}_q$.*

Replace the term $R_k(x)\tau^k \frac{dx}{y^\ell}$ in ω with $\left(A_k(x) + \frac{r}{r(k-1)+\ell}B_k'(x)\right)\tau^{k-1}\frac{dx}{y^\ell}$.

end for

Step 2: Reduce the degree in x, that is, transform $S(x)\frac{dx}{y^\ell}$ to $T(x)\frac{dx}{y^\ell}$, with $\deg(T) \leq d-2$:

$T := S$;

$m := \deg(T)$;

while $m \geq d-1$ **do**

$\widetilde{T} := r(m-d+1)x^{m-d}\overline{f}(x) + (r-\ell)x^{m-d+1}\overline{f}'(x)$;

$\widetilde{T} := \frac{LC(T)}{LC(\widetilde{T})}\widetilde{T}$; *//LC is the Leading Coefficient*

$T := T - \widetilde{T}$;

$m := \deg(T)$;

end while

return $T(x)\frac{dx}{y^\ell}$;

(using the Coppersmith–Winograd algorithm, for example). We let s be such that the permutation $j \mapsto qj \bmod r$ of $\{1, \cdots r-1\}$ is a product of s cycles of lengths $c_1, c_2, \cdots, c_s$.

Proposition 5.1. *With the notation above, the Weil polynomial of a cyclic cover $\mathcal{C} : y^r = f(x)$ defined over $\mathbb{F}_{p^n}$, with f of degree d can be computed using Algorithm 1 in time $\widetilde{O}\left(pn^3d^4r^3 + n^2rd^{\nu+1}(\sum_{i=1}^s c_i^\nu)\right)$ elementary operations and space $\widetilde{O}\left(pn^3d^3r^2 + n^2d^3r(\sum_{i=1}^s c_i^2)\right)$ bits of memory.*

Proof. We first describe the bit size of the different objects. An element of $\mathbb{Z}_q$ is represented by a polynomial of degree $n-1$ with coefficients in $\mathbb{Z}_p$ truncated to precision N, so it has size $O(nN\log(p)) = \widetilde{O}(nN)$. An element of $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$ is represented by a degree $\mu = O(pN)$ polynomial in τ whose coefficients are polynomials over $\mathbb{Z}_q$ of degree less than d, so it has size $\widetilde{O}\left(pndN^2\right)$.

Thanks to Schönhage and Strassen, the multiplication between two integers of bit-size k is $\widetilde{O}(k)$ elementary operations [**vzGG03**]. Hence, the complexity of multiplying two elements of $\mathbb{Z}_q$ is $\widetilde{O}(nN)$ elementary operations and the cost of multiplying two elements of $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$ is $\widetilde{O}\left(pndN^2\right)$ elementary operations.

Recall that we normalize the elements of $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$ at each step using the equation of the curve, so we need to calculate the complexity of the normalization. This procedure is described in [**GG03, vzGG03**], so we will not go into further detail here. If we want to normalize $Q(x)\tau^k$, then it costs $\widetilde{O}(\deg(Q))$ operations in $\mathbb{Z}_q$ so the complexity of the normalization step is $\widetilde{O}(nN\deg(Q))$ elementary operations.

We compute the Frobenius substitution on $\mathbb{Z}_q$ by a Newton iteration and Hörner's method: if $z = \sum_{k=0}^{n-1} z_k t^k$ is an element of $\mathbb{Z}_q$ then $z^\sigma = \sum_{k=0}^{n-1} z_k t^{\sigma k}$ where t^σ is computed using a Newton iteration. The complexity of the Newton algorithm is determined by the last iteration, which costs $O(n)$ operations in $\mathbb{Z}_q$, that is $\widetilde{O}\left(n^2 N\right)$ elementary operations. Hörner's method costs $O(n)$ operations in $\mathbb{Z}_q$, that is, $\widetilde{O}\left(n^2 N\right)$ elementary operations. Hence, we compute the Frobenius substitution on $\mathbb{Z}_q$ in $\widetilde{O}\left(n^2 N\right)$ elementary operations.

In Step 2 of Algorithm 1, we compute the inverse of $\mathcal{F}(y)$ by a Newton iteration. We first compute R, by computing the polynomial $\mathcal{F}(\overline{f}(x)) - \overline{f}(x)^p$ of degree pd. We then normalize R which costs $\widetilde{O}\left(nN \deg(R)\right) = \widetilde{O}\left(nNpd\right)$ elementary operations. We then apply the Newton algorithm to R in order to compute S as the inverse of its r-th root. The complexity of the Newton algorithm is a constant times the cost of its last iteration which consists of some multiplications between two elements of $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$ and a normalization. The cost of the Newton iteration is therefore $\widetilde{O}\left(pndN^2\right)$ elementary operations. So the cost of Step 2 is $\widetilde{O}\left(pndN^2\right)$ elementary operations. This step requires $\widetilde{O}\left(pndN^2\right)$ bits of memory.

In Step 3 of Algorithm 1, we first compute S^j which costs $\widetilde{O}\left(pndN^2\right)$ elementary operations. We then apply a normalization to $px^{p(i+1)-1}S^j$ which costs $\widetilde{O}\left(pdnN\right)$ elementary operations in the worst case. Then we perform the reduction steps using Algorithm 2. In Step 1 of Algorithm 2, we do μ iterations: each time we compute A_k and B_k using the extended Euclidean algorithm, which costs $O(d)$ operations in $\mathbb{Z}_q$, that is $\widetilde{O}\left(dnN\right)$ elementary operations. Then we replace the term in ω of highest degree in τ by performing d additions in $\mathbb{Z}_q$, which costs $\widetilde{O}\left(ndN\right)$ elementary operations. So the cost of Step 1 of Algorithm 2 is $\widetilde{O}\left(\mu dnN\right) = \widetilde{O}\left(pdnN^2\right)$ elementary operations. During the second Step of Algorithm 2, we do $\deg(S) \leq dp$ iterations which consists of d operations of elements of $\mathbb{Z}_q$, so the cost of Step 2 in Algorithm 2 is $\widetilde{O}\left(dp \times d \times nN\right) = \widetilde{O}\left(pd^2nN\right)$ elementary operations and it requires $\widetilde{O}\left(pndN^2\right)$ bits of memory.

The complexity of reducing a differential form with Algorithm 2 is therefore $\widetilde{O}\left(pdnN(N+d)\right)$. As we need to reduce $O(rd)$ differential forms, the complexity of Algorithm 2 is $\widetilde{O}\left(pd^2nrN(N+d)\right)$ elementary operations, and since we reduce differential forms one by one, Algorithm 2 requires $\widetilde{O}\left(pndN^2\right)$ bits of memory. Putting everything together, the complexity for Step 3 of Algorithm 1 is $\widetilde{O}\left(pd^2nrN(N+d)\right)$ elementary operations and it requires $\widetilde{O}\left(pndN^2\right)$ bits of memory.

In Step 4 of Algorithm 1, we compute the matrix of the q-th power Frobenius. Recall that $M_{\mathcal{F}}$ is a block diagonal matrix of s blocks matrices $M_{\mathcal{F},i}$ of size $c_i(d-1) \times c_i(d-1)$ for $1 \leq i \leq s$. Note that $M_{\mathcal{F},i}$ is itself a block matrix, composed of c_i blocks of size $(d-1) \times (d-1)$, with only one non zero block on each row partition and column partition. The matrix M is also a block diagonal matrix: its blocks are the norms

$$M_i = M_{\mathcal{F},i} \cdot M_{\mathcal{F},i}^{\sigma} \cdots M_{\mathcal{F},i}^{\sigma^{n-1}}.$$

Each of the M_i can be computed using Hörner's method (and the sub-block structure of $M_{\mathcal{F},i}$); this costs $\widetilde{O}\left(nc_i d^\nu\right)$ operations in $\mathbb{Z}_q$ and requires $\widetilde{O}\left(c_i d^2 nN\right)$ bits of memory. The total cost of computing M is therefore $\widetilde{O}\left(n\left(\sum_{i=1}^{s} c_i\right) d^\nu nN\right)$, that is, $\widetilde{O}\left(n^2 r d^\nu N\right)$ elementary operations (because $\sum_{i=1}^{s} c_i = r$) and requires $\widetilde{O}\left(rd^2 nN\right)$ bits of memory.

We then compute the characteristic polynomial of M which is the product of the characteristic polynomials χ_{M_i} of the M_i. The complexity of computing the characteristic polynomial of a square matrix of size k over a ring $\mathcal{R}$ is $\widetilde{O}(k^\nu)$ operations in $\mathcal{R}$. Hence, computing χ_{M_i} costs $\widetilde{O}((c_i d)^\nu)$ operations in $\mathbb{Z}_q$. So, computing χ_M costs $\widetilde{O}\left(\left(\sum_{i=1}^s c_i^\nu\right) d^\nu nN\right)$ elementary operations and requires $\widetilde{O}\left(\left(\sum_{i=1}^s c_i^2\right) d^2 nN\right)$ bits of memory.

The global cost of Step 4 is therefore $\widetilde{O}\left(\left(nr + \left(\sum_{i=1}^s c_i^\nu\right)\right) d^\nu nN\right)$ elementary operations and requires $\widetilde{O}\left(\left(\sum_{i=1}^s c_i^2\right) d^2 nN\right)$ bits of memory.

In Step 5 of Algorithm 1, we compute the polynomial U of degree $\delta - 1$ over the integers. This corresponds to multiplying at most $\delta - 1$ binomials with coefficients no larger that p^n. So this costs $\widetilde{O}(n\delta)$ elementary operations. We then divide the polynomial obtained in Step 5 by U, so Step 5 costs the equivalent of $O(g) = O(rd)$ operations in $\mathbb{Z}_q$, that is $\widetilde{O}(rdnN)$ elementary operations and requires $\widetilde{O}(rdN)$ bits of memory.

The total complexity of our algorithm is therefore

$$\widetilde{O}\left(pndN^2 + pd^2nrN(N+d) + \left(nr + \left(\sum_{i=1}^s c_i^\nu\right)\right) d^\nu nN + rdnN\right)$$

elementary operations and

$$\widetilde{O}\left(pndN^2 + d^2nN\left(\sum_{i=1}^s c_i^2\right)\right)$$

bits of memory. Theorem 4.3 tells us that $N = \widetilde{O}(ng) = \widetilde{O}(nrd)$, so the complexity of our algorithm is

$$\widetilde{O}\left(pn^3d^4r^3 + n^2rd^{\nu+1}\left(\sum_{i=1}^s c_i^\nu\right)\right)$$

elementary operations and

$$\widetilde{O}\left(pn^3d^3r^2 + n^2d^3r\left(\sum_{i=1}^s c_i^2\right)\right)$$

bits of memory. □

Remark 5.2. Note that if $q \equiv 1 \bmod r$, then $s = r - 1$ and $c_i = 1$ for $1 \leq i \leq s$, and hence the complexity of Algorithm 1 is $\widetilde{O}\left(pn^3d^4r^3\right)$, which is the better case.

Remark 5.3. We can improve the complexity of this algorithm in larger characteristic to $\widetilde{O}\left(\sqrt{p}n^3d^4r^3 + n^2rd^{\nu+1}(\sum_{i=1}^s c_i^\nu)\right)$, by applying Harvey's improvements [**Har07**] to Kedlaya's algorithm which were extended to superelliptic curves by Minzlaff [**Min10**]. Indeed, our algorithm is entirely compatible with Minzlaff's improvements and we can apply them to our algorithm. These improvements consist of two major key points. First, they use a different representation for the images of differential forms under the action of Frobenius: in Kedlaya's algorithm, we use an approximation by series whose number of terms is linear in p whereas Harvey's improvements use a different series approximation which does not depend on p. Second, these improvements reduce the complexity of the reduction algorithm, which is the major step, by solving a linear recurrence using the Bostan–Gaudry–Schost algorithm [**BGS07**].

6. Bounds on precision

In this section, we give a proof of Theorem 4.3. In order to compute the Weil polynomial exactly, we need to take sufficient precision. The Weil bounds give us a minimal bound N_0, but this bound is not sufficient since the divisions in the reduction algorithm (Algorithm 2) induce a loss of precision. Proposition 6.1 estimates the loss of precision during the first step of Algorithm 2, and Proposition 6.2 estimates the loss of precision during the second step of Algorithm 2.

PROPOSITION 6.1. *If R is a polynomial defined over $\mathbb{Z}_q$ with degree less than d, then the first step in Algorithm 2 transforms $R(x)\tau^k \frac{dx}{y^\ell}$, with $k > 1$, into $S(x)\frac{dx}{y^\ell}$, where S is a polynomial defined over $\mathbb{Q}_q$ with degree less than d. Moreover, the coefficients of S have denominator bounded by $p^{\lfloor \log_p(r(k-1)+\ell)\rfloor}$.*

PROOF. We follow the approach of [**Ked01**, Lemma 2] and [**Edi03**, Lemma 4.3.4].

During the first step of Algorithm 2, we apply (3.8) several times. Hence, divisions by $r(i-1)+\ell$ are done, which corresponds to negative powers of y appearing during this step, for each $1 \leq i \leq k$, and positive powers of p may occur in denominators. It is then natural to look at what happens at the poles of y^{-1}, that is the points $P_i = (\alpha_i, 0)$, with α_i a root of $\overline{f}$ in $\mathbb{Z}_q$ (note that α_i is a simple root of $\overline{f}$), in order to deduce what happens globally.

Let $Q = \sum_{j=1}^{k-1} Q_j(x)\frac{\tau^j}{y^\ell}$ be such that $R(x)\tau^k \frac{dx}{y^\ell} = S(x)\frac{dx}{y^\ell} + dQ$, with Q_j defined over $\mathbb{Q}_q$ of degree less than d, for any $1 \leq j < k$. As α_i is a simple root of $\overline{f}$, the function y is a uniformizing parameter for the local ring $\mathcal{O}_{P_i}$, that is, the ring of functions on $\mathcal{C}$ regular at P_i. Thus, the weak completion $\mathcal{O}_{P_i}^\dagger$ of $\mathcal{O}_{P_i}$ is $\mathbb{Z}_q[[y]]$. We can then write dQ and Q as series:

$$dQ = \sum_{j \geq -(r(k-1)+\ell+1)} c_j y^j dy$$

and

$$Q = \sum_{j \geq -(r(k-1)+\ell+1)} \frac{c_j}{j+1} y^{j+1}.$$

As c_j coincides with the corresponding coefficient of $R(x)\tau^k \frac{dx}{y}$ when $j < 0$, it lies in $\mathbb{Z}_q$. Hence, if we set $m = p^{\lfloor \log_p(r(k-1)+\ell)\rfloor}$, then $m\frac{c_j}{j+1}$ is integral (ie in $\mathbb{Z}_q$) for $j < 0$. Evaluating the coefficient of $y^{-(r(k-1)+\ell)}$ at a pole P_i of y^{-1} in the expression $Q = \sum_{j=1}^{k-1} Q_j(x)\frac{\tau^j}{y^\ell}$ gives

$$Q_{k-1}(\alpha_i) = \frac{c_{-(r(k-1)+\ell+1)}}{-(r(k-1)+\ell)},$$

so $mQ_{k-1}(\alpha_i)$ is integral. As this statement is independent of the point P_i chosen, it holds for any $1 \leq i \leq d$. Thus, mQ_{k-1} (of degree less than d) is integral at each of the d distinct poles of y^{-1} and it follows that mQ_{k-1} is a polynomial defined over $\mathbb{Z}_q$. The same argument applied to $Q - Q_{k-1}\tau^{k-1}$ gives that mQ_{k-2} is a polynomial defined over $\mathbb{Z}_q$. By induction, all the mQ_k are defined over $\mathbb{Z}_q$. It follows that mQ is integral and then mS is as well because $S(x)\frac{dx}{y} = R(x)\frac{dx}{y} - dQ$. □

PROPOSITION 6.2. *If S is a polynomial defined over $\mathbb{Z}_q$ with degree $m \geq d-1$, then the second step in Algorithm 2 transforms $S(x)\frac{dx}{y^\ell}$ into $T(x)\frac{dx}{y^\ell}$, where T is a polynomial defined over $\mathbb{Q}_q$ with degree less than $d-1$. The coefficients of T have denominator bounded by $p^{\lfloor \log_p(\frac{r(m+1)-\ell d}{\delta})\rfloor}$.*

PROOF. We follow the approach of [**Edi03**, Lemma 4.3.5].

During the second step of Algorithm 2, we apply (3.9) several times. As we divide by the leading coefficient of the polynomial given in (3.9), positive powers of p may occur at the denominators, which corresponds to positive powers of x appearing in this step. Hence we study what happens at the poles of x, that is the points at infinity $P_{\infty,k} = [1 : \zeta_r^k : 0]$ of $\mathcal{C}$, with $1 \leq k \leq \delta$.

Let $v_{\infty,k}$ denote the valuation at $P_{\infty,k}$. Let $Q = \sum_{i=d-1}^{m} r a_i x^{i-d+1} y^{r-\ell}$ be such that $S(x)\frac{dx}{y^\ell} = T(x)\frac{dx}{y^\ell} + dQ$. Then $v_{\infty,k}(x) = -\frac{r}{\delta}$, $v_{\infty,k}(y) = -\frac{d}{\delta}$ so $v_{\infty,k}(dx) = -\frac{r+\delta}{\delta}$. Moreover, $v_{\infty,k}\left(\frac{dx}{y^\ell}\right) = \frac{\ell d - r - \delta}{\delta}$ so

$$v_{\infty,k}(Q) \geq \frac{\ell d - r(m+1)}{\delta}$$

and

$$v_{\infty,k}\left(T\frac{dx}{y^\ell}\right) \geq \frac{\ell d - r(d-1) - \delta}{\delta}.$$

Let z_k be a local uniformizer at $P_{\infty,k}$, so that $\mathcal{O}_{P_{\infty,k}}$ is $\mathbb{Z}_q[[z_k]]$. Then we have the following expansion in $\mathbb{Z}_q[[z_k]]$:

$$dQ = \sum_{j \geq \frac{\ell d - r(m+1)}{\delta} - 1} c_j z_k^j dz_k$$

and

$$Q = \sum_{j \geq \frac{\ell d - r(m+1)}{\delta} - 1} \frac{c_j}{j+1} z_k^{j+1}.$$

Let $m = p^{\lfloor \log_p\left(\frac{r(m+1)-\ell d}{\delta}\right)\rfloor}$. As $v_{\infty,k}(T\frac{dx}{y^\ell}) \geq -\frac{r(d-1)-\ell d+\delta}{\delta}$, then the c_j are in $\mathbb{Z}_q$ for $j \leq -(r(d-1) - \ell d + \delta)/\delta - 1 = -(d(r-\ell) + 2\delta - r)/\delta$ since they coincide with the coefficients of S, so $m\frac{c_j}{j+1}$ is in $\mathbb{Z}_q$ for $j \leq -\frac{(d(r-\ell)+2\delta-r)}{\delta}$.

As all the $v_{\infty,k}(x^i \frac{dx}{y^\ell})$ are distinct and less than $\frac{\ell d - r - \delta - r(d-1)}{\delta} \leq -\frac{d(r-\ell)+2\delta-r}{\delta}$ for $d-1 \leq i \leq m$ and $\ell > 0$, it follows that all the terms mQ_j are in $\mathbb{Z}_q[x]$.

Since the previous statements are independent of k, they hold for any point at infinity of $\mathcal{C}$. Hence mQ is in $\mathbb{Z}_q[x]$, and mT is in $\mathbb{Z}_q[x]$ too. □

The two previous propositions estimate the loss of precision resulting from the reductions. Recall that we are working with p-adic elements up to precision N, and that every element of $\mathcal{A}^\dagger$ is an overconvergent series, that is a series whose coefficients have p-adic valuation which grows at least linearly. This means that for any $0 \leq i \leq d-2$ and $1 \leq j \leq r-1$, $\mathcal{F}\left(\frac{x^i dx}{y^j}\right)$ is a power series of the form $\sum_{k \geq 0} F_k \tau^k$ and there exists μ such that $v_p(F_k)$ is greater than p^N, for all $k > \mu$. Thus in practice, all the computed series are in fact polynomials of degree at most μ in τ.

PROOF OF THEOREM 4.3. The Weil bounds state that, if we put

$$N_0 = \left\lceil \log_p\left(2\binom{2g}{g} q^{g/2}\right)\right\rceil,$$

then N_0 is the minimal precision we have to take to compute the Weil polynomial exactly. We also have to take into account all the digits lost by the divisions done during the reduction steps.

Let N be the total precision we must take to compute the zeta function exactly, and N_1 the intermediate precision we must take to do the first reduction step up to precision N_1. We will determine N as a function of N_1 and N_1 as a function of N_0 to recover N as a function of N_0.

Let μ denote the integer such that the $v_p(F_k)$ are greater than N for $k > \mu$ (they are zero modulo p^N).

Using the expression of the action of Frobenius on vectors of B given by (3.7), we find that the integer μ we want to determine is such that $\frac{k-a}{p} + 1 \geq N$ for $k > \mu$ and $\frac{\mu-a}{p} + 1 < N$, so

$$\mu = p(N-1) + a - 1.$$

Note that $a < p$, since a is the quotient in the division of jp by r, and that $j \leq r-1$. Hence, $\mu < pN - 1$.

To determine N, let us have a look at what happens during the reductions. Proposition 6.1 says that we lose $\lfloor \log_p (r(k-1)+\ell) \rfloor$ digits of precision during the first step of Algorithm 2, when we reduce $Q_k \tau^k \frac{dx}{y^\ell}$ with $\deg(Q_k) < d$. We want to take N as small as possible such that after this first step of reduction, there remains N_1 digits of precision for the second step of Algorithm 2, that is, such that

$$\frac{k-a}{p} + 1 - \lfloor \log_p (r(k-1)+\ell) \rfloor \geq N_1 \quad \text{for } k > \mu \tag{6.1}$$

(here N appears in the expression of μ).

The function $g : [\mu+1\ ,\ +\infty) \to \mathbb{R}$ mapping k to the left hand side of Inequality (6.1) is strictly increasing; so we take the smallest N such that $g(\mu+1) \geq N_1$. We find that N is the minimal integer satisfying

$$N - \lfloor \log_p (p(rN-1) - r) \rfloor \geq N_1. \tag{6.2}$$

In order to determine N_1, consider what happens during the second step of Algorithm 2: the terms contributing to the loss of precision during this step are those with degree 0 in τ, that is, the polynomials with degree in x greater than $d-2$. For $1 \leq i \leq d-2$ and $1 \leq j \leq r$, we have

$$\mathcal{F}\left(x^i \frac{dx}{y^j}\right) = \sum_{k \geq 0} \binom{-j/r}{k} p^{k+1} E^k x^{p(i+1)-1} \tau^{pk+a} \frac{dx}{y^\ell}$$

and $i = d-2$ at worst, so $\deg(E^k x^{p(d-1)-1}) \leq k(pd-1) + p(d-1) - 1$. As $\frac{k(pd-1)+p(d-1)-1}{d} < (k+1)p$, we can write

$$\binom{-j/r}{k} p^{k+1} E^k x^{p(i+1)-1} \tau^{pk+a} = \sum_{a-p<j<pk+a} c_{j,k} \tau^j,$$

so the degree in x of the constant term of $\mathcal{F}\left(x^i \frac{dx}{y}\right)$ is at most $d\ (p\ -\ a)$.

Proposition 6.2 says that the number of digits lost during this second step is

$$\left\lfloor \log_p \left(\frac{r\,(d(p-a)+1) - \ell d}{\delta} \right) \right\rfloor.$$

Since $jp = ar + \ell$, we lose $\left\lfloor \log_p \left(\frac{dp(r-j)+r}{\delta} \right) \right\rfloor \leq \left\lfloor \log_p \left(\frac{dp(r-1)+r}{\delta} \right) \right\rfloor$ digits. Hence,

$$N_1 = N_0 + \left\lfloor \log_p \left(\frac{r(dp+1) - d}{\delta} \right) \right\rfloor. \tag{6.3}$$

Let us put these two parts together. Combining Equations (6.2) and (6.3), we should take

$$\begin{aligned} N &= \min_{n \in \mathbb{N}} \left\{ n - \left\lfloor \log_p \Big(p(rn-1) - r \Big) \right\rfloor \geq N_1 \right\} \\ &= \min_{n \in \mathbb{N}} \left\{ n - \left\lfloor \log_p \Big(p(rn-1) - r \Big) \right\rfloor \geq N_0 + \left\lfloor \log_p \left(\frac{dp(r-1)+r}{\delta} \right) \right\rfloor \right\}. \end{aligned}$$

□

7. The choice of the set of differentials

In Step 5 of Algorithm 1 we compute the norm M of the matrix of Frobenius $M_{\mathcal{F}}$ with respect to B, and in Step 6 we compute its characteristic polynomial $\chi(t)$. If $M_{\mathcal{F}}$ has coefficients with denominators (coefficients in $\mathbb{Q}_q \setminus \mathbb{Z}_q$), then it is difficult to control the valuation of these denominators in the norm, and worse, we have to enlarge the precision bounds to recover the Weil polynomial exactly by a number of digits that is hard to estimate.

So, it would be ideal if $M_{\mathcal{F}}$ was guaranteed to have coefficients in $\mathbb{Z}_q$. Proposition 7.1 tells us whether $M_{\mathcal{F}}$ has integral coefficients or not.

PROPOSITION 7.1. *Let $0 \leq i \leq d-2$ and $1 \leq j \leq r-1$.*

- *The first step in Algorithm 2, applied to $\mathcal{F}(x^i \frac{dx}{y^j})$, computes a differential form whose coefficients have denominators of valuation bounded by $\lfloor \log_p(r) \rfloor$.*
- *The second step in Algorithm 2, applied to $\mathcal{F}(x^i \frac{dx}{y^j})$, computes a differential form whose coefficients have denominators of valuation bounded by $\left\lfloor \log_p\left(\frac{2g+(\delta-2)}{\delta}\right)\right\rfloor$.*

PROOF. In this proof, we follow the approach of [**Har12**, Lemma 3.4].

Recall that

$$\begin{aligned}\mathcal{F}\left(x^i \tfrac{dx}{y^j}\right) &= x^{p(i+1)-1} \textstyle\sum_{k\geq 0} \binom{-j/r}{k} E^k p^{k+1} \tau^{pk+a} \frac{dx}{y^\ell} \\ &= x^{p(i+1)-1} \textstyle\sum_{k\geq a} p^{\frac{k-a}{p}+1} Q_k(x) \tau^k \frac{dx}{y^\ell},\end{aligned}$$

where $jp = ar + \ell$. Lemma 6.1 shows that after the first step in Algorithm 2 the valuation in p of the coefficients have the form:

$$k + 1 - \left\lfloor \log_p\left(r(pk+a-1)+\ell\right)\right\rfloor \geq k - \left\lfloor \log_p\left(kr+j\right)\right\rfloor. \tag{7.1}$$

Let g be the function defined on $[0, +\infty)$ by $g(x) = k - \log_p(kr+j)$. Since g is strictly increasing, the right hand side is maximal when $k = 0$, which implies that the left hand side in Inequality 7.1 is greater than $\lfloor \log_p(j) \rfloor \leq \lfloor \log_p(r) \rfloor$.

Now consider the terms $p^{\frac{k-a}{p}+1} Q_k(x) x^{p(i+1)-1} \tau^k \frac{dx}{y^\ell}$ appearing in the second step of Algorithm 2. After normalizing the coefficients, each term will be expressible in the form $S\frac{dx}{y^\ell}$, with

$$\deg(S) = p(i+1) - 1 + \deg(Q_k) - dk.$$

After the second step in Algorithm 2, the denominators are bounded by

$$A = \left\lfloor \log_p\left(\frac{rp(i+1) + r\deg(Q_k) - rkd - \ell d}{\delta}\right)\right\rfloor - 1 - \frac{k-a}{p}$$

Since $Q_a = 1$ and $\deg(Q_k) < d$ for any $k > a$, this expression is maximal when $k = a$, which gives

$$A \leq \left\lfloor \log_p\left(\frac{rp(i+1) - (ra+\ell)d}{\delta}\right)\right\rfloor - 1.$$

As $jp = ar + \ell$, we can express the right hand side of the inequality as $\left\lfloor \log_p\left(\frac{r(i+1)-jd}{\delta}\right)\right\rfloor$, which is less than $\left\lfloor \log_p\left(\frac{d(r-1)-r}{\delta}\right)\right\rfloor$. Relation 2.2 allows us to replace $d(r-1)$ with $2g + (\delta - 2)$.

Hence, the denominators of the coefficients of $M_{\mathcal{F}}$ are bounded by

$$\left\lfloor \log_p\left(\frac{2g - (\delta-2)}{\delta}\right)\right\rfloor.$$

□

This proposition tells us that if $d > r$, then the denominators mostly come from the second step of Algorithm 2. In this case, if $p \geq \frac{2g-(\delta-2)}{\delta}$, Algorithm 1 gives us a matrix $M_{\mathcal{F}}$ with integral coefficients, while when $p < \frac{2g-(\delta-2)}{\delta}$ they have denominators bounded by $\left\lfloor \log_p\left(\frac{2g-(\delta-2)}{\delta}\right)\right\rfloor$. So if we want to minimize these denominators, we should find a set of differentials which avoids this second step in Algorithm 2.

7.1. Another set of differentials which avoids the second step in Algorithm 2. Consider the set of differentials

$$B' = \left\{ x^i \frac{dx}{y^{r+j}} \mid i \in [0, \dots, d-2], j \in [1, \dots, r-1] \right\}.$$

The reduction formulae in §3 tells us that B' spans $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$. We will show in Proposition 7.4 that $\langle B' \rangle$ decomposes into a direct sum of two subspaces including $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$, so we can recover the Weil polynomial of $\mathcal{C}$ from the action of Frobenius acting on B'. Moreover, the matrix of Frobenius with respect to B' has the same block structure as the matrix of Frobenius with respect to B.

We will show in Proposition 7.4 that when $\delta = 1$, then B' is a basis of $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$. Thus, we can do the computations with B' instead of B in the Gaudry–Gürel algorithm.

The following theorem tells us that doing the computations with B' avoids the second step in Algorithm 2.

THEOREM 7.2. *Let $0 \leq i \leq d-2$ and $1 \leq j \leq r-1$. The first step in Algorithm 2, applied to $\mathcal{F}\left(x^i \frac{dx}{y^{j+r}}\right)$, gives a form which is a linear combination of elements of B' and whose coefficients have denominator bounded by $p^{\lfloor \log_p(2r-1) \rfloor}$.*

PROOF. In this proof, we follow the approach of [**Har12**, Lemma 3.4].

Let $0 \leq i \leq d-2$, $1 \leq j \leq r-1$ and $\varepsilon = 0$ or 1. Using the same calculation as we did to get (3.6), we have

$$\mathcal{F}\left(x^i \frac{dx}{y^{\varepsilon r+j}}\right) = x^{p(i+1)-1} \sum_{k \geq 0} \binom{-(\varepsilon r+j)/r}{k} p^{k+1} E^k(x) \tau^{p(k+\varepsilon)+a-\varepsilon} \frac{dx}{y^{\varepsilon r+\ell}},$$

where $E = \frac{\mathcal{F}(\overline{f}(x)) - \overline{f}(x)^p}{p}$ and $jp = ar + \ell$. After normalizing (using the equation of $\mathcal{C}$ to remove all the terms with degree in x greater than $\deg(f)$), and using the fact that $Q_{a+\varepsilon(p-1)} = 1$, we have

$$\mathcal{F}\left(x^i \frac{dx}{y^{\varepsilon r+j}}\right) = \sum_{k \geq k_0} p^{\frac{k-a+\varepsilon}{p}+1-\varepsilon} R_k(x) \tau^k \frac{dx}{y^{\varepsilon r+\ell}}, \tag{7.2}$$

with $k_0 \geq a + \varepsilon(p-1) - \left\lfloor \frac{p(i+1)-1}{d} \right\rfloor$.

When $\varepsilon = 1$, we have $k_0 \geq a + (p-1) - \left\lfloor \frac{p(i+1)-1}{d} \right\rfloor$. The right hand side is minimal when $i = d-2$, with value $a + p - 1 - (p-1) = a \geq 0$. This means that when $\varepsilon = 1$, the only term which may be reduced by the second step of Algorithm 2 is the first one, which has degree 0. So the first step in Algorithm 2, applied to $\mathcal{F}\left(x^i \frac{dx}{y^{j+r}}\right)$ gives a form which is a linear combination of elements of B'.

Consider the terms $\binom{-(r+j)/r}{k} p^{k+1} E^k \tau^{p(k+1)+a-1}$ appearing in the first step of Algorithm 2. Lemma 6.1 shows that after the first step in Algorithm 2 the valuation in p of the coefficients is

$$k + 1 - \left\lfloor \log_p\left(r(p(k+1) + a - 2) + \ell\right) \right\rfloor \geq k - \left\lfloor \log_p\left(r(k+1) + j\right) \right\rfloor$$

An easy calculation shows that the right hand side term is maximal when $k = 0$ and has value $\lfloor \log_p(r+j) \rfloor \leq \lfloor \log_p(2r-1) \rfloor$. □

If $p \geq 2r$, then the computations done in Algorithm 1 using B' will give a matrix with integral coefficients. If however $p < 2r$, then the matrix computed with respect to B' will have coefficients with denominators bounded by $p^{\lfloor \log_p(2r-1) \rfloor}$.

At the end of this subsection, we will compare the two sets of differentials B and B' and provide a criterion to recommend which one to use.

As there is no second step of reduction when we use B' in the computations, we can slightly reduce the bounds on precision as follows.

THEOREM 7.3. *In order to compute the Weil polynomial of $\mathcal{C}$ exactly using Algorithm 1 with the basis B', we can take*

$$N = \min_{n \in \mathbb{N}} \left\{ n - \left\lfloor \log_p \left(pr(n+1) - 3r \right) \right\rfloor \geq N_0 \right\}$$

where

$$N_0 = \left\lceil \log_p \left(2 \binom{2g}{g} q^{g/2} \right) \right\rceil.$$

PROOF. We follow the approach of the proof of Theorem 4.3.

Using (7.2), we find that

$$\mu = pN + a - 2.$$

As the second step in Algorithm 2 is not executed, we only need to take into account the loss of precision resulting from the first step in Algorithm 2. So we want to find N as small as possible such that after the first step of reduction, there remains N_0 digits of precision, that is

$$\tfrac{k-a+1}{p} - \left\lfloor \log_p \left(r(k-1) + \ell \right) \right\rfloor \geq N_0 \quad \text{for } k > \mu \tag{7.3}$$

(here N appears in the expression of μ). The function $g : [\mu + 1\ ,\ +\infty) \to \mathbb{R}$ which maps k to the left hand side of Inequality (6.1) is strictly increasing; so we take the smallest N such that $g(\mu+1) \geq N_1$. We find that N is the minimal integer satisfying

$$N - \left\lfloor \log_p \left(pr(N+1) - 3r \right) \right\rfloor \geq N_0, \tag{7.4}$$

which ends the proof. □

Finally we observe that if $p \geq 2r$, then using B' instead of B in the computations is much better for two reasons. First, it always lead to an integral matrix. Second, as the second step of Algorithm 2 is not executed, we do fewer operations and at a lower precision (because we save near $\left\lfloor \log_p \left(\frac{dp(r-1)+r}{\delta} \right) \right\rfloor$ digits of precision) so it slightly decrease the complexity of Algorithm 1.

Otherwise, it is not always possible to have a matrix with integral coefficients and we have to check if

$$\max \left(\lfloor \log_p(r) \rfloor, \left\lfloor \log_p \left(\frac{2g - (\delta - 2)}{\delta} \right) \right\rfloor \right)$$

is greater than

$$\lfloor \log_p(2r-1) \rfloor.$$

If it is the case, then we use B' and if not, we have to choose between B and B', taking into account that we will have to enlarge the bounds on precision (which are smaller using B') due to the computation of $M = M_{\mathcal{F}} \cdot M_{\mathcal{F}}^{\sigma} \cdots M_{\mathcal{F}}^{\sigma^{n-1}}$ and its characteristic polynomial.

7.2. How does the use of B' change Algorithm 1 ? Using B' instead of B does not significantly change Steps 1 to 5 of our algorithm, where we compute the action of Frobenius on cohomology: the operations are the same, only some powers of p and some indexes in the expressions change. However, the extra factor U in Step 6 of Algorithm 1 may be slightly different.

PROPOSITION 7.4. *Let $\eta : \langle B' \rangle \to H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$ be the map sending differentials to their classes in cohomology.*

- *If $\delta = 1$, then η is an isomorphism.*
- *Otherwise, η has a $\delta - 1$-dimensional kernel stable under the action of Frobenius, and there exists a basis W of $\ker(\eta)$ such that the matrix of the q-th power Frobenius with respect to W is a generalized permutation matrix: that is, in the form DP where D is a diagonal matrix and P is a permutation matrix.*

PROOF. We follow the approach of [**Har12**, Proposition 3.5].

The reduction formulae in §3 tells us that B' spans $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$ so η is surjective and η has a kernel of dimension

$$\dim(\mathrm{span}(B')) - \dim(H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)) = \delta - 1.$$

It follows that if $\delta = 1$, then η is an isomorphism.

Otherwise, let

$$\omega = \sum_{j=1}^{r-1} \left(\sum_{i=0}^{d-2} \lambda_{i,j} x^i \right) \frac{dx}{y^{j+r}} = \sum_{j=1}^{r-1} V_j(x) \frac{dx}{y^{j+r}}$$

be a non zero element of $\ker(\eta)$, that is, such that $\omega \equiv 0$ in $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$. We want to determine $\ker(\eta)$, that is, to describe ω explicitly.

Using the first reduction formula, we get

$$0 \equiv \omega \equiv \sum_{j=1}^{r-1} \left(A_j + \frac{r}{j} B'_j \right) \frac{dx}{y^j}, \text{ where } V_j = A_j f + B_j f'.$$

As the degree of V_j is lower than $d-1$ for $1 \leq j \leq r-1$, the degrees of A_j and B_j are lower to $d-1$ too, and there is no second reduction, so we have expressed ω as a linear combination of elements of B. Since the elements of B are linearly independent, it follows that

$$A_j + \frac{r}{j} B'_j = 0 \text{ for all } 1 \leq j \leq r-1,$$

so ω has the form

$$\omega = \sum_{j=1}^{r-1} \left(-\frac{r}{j} B'_j f + B_j f' \right) \frac{dx}{y^{r+j}}.$$

In order to describe B_j for $1 \leq j \leq r-1$, let j be an integer such that $1 \leq j \leq r-1$. If we write $B_j = a_{b_j} x^{b_j} + \cdots$ with $b_j \geq 0$ and $a_{b_j} \neq 0$, then the leading term in V_j is $\left(d - \frac{r}{j} b_j \right) a_b x^{d-1+b_j}$. Since $\deg(V_j) \leq d-2$, we have that $b_j = \frac{jd}{r}$. Normalizing B_j so that its leading coefficient is 1, it follows easily that its lower coefficients are completely determined inductively by the condition on $\deg(V_j)$. Hence

$$\omega = \sum_{1 \leq j \leq r-1, \frac{r}{\delta} | j} \lambda_j \omega_j,$$

where

$$\omega_j = V_j \frac{dx}{y^{r+j}}$$

and j is a multiple of $\frac{r}{\delta}$: that is, where $j = \alpha\frac{r}{\delta}$ with $1 \leq \alpha \leq \delta - 1$. We may take

$$\omega_j = \left(B_j f' - \frac{r}{j} B_j' f\right) \frac{dx}{y^{r+j}} = d\left(-\frac{r}{j}\frac{B_j}{y^j}\right),$$

with $\deg(B_j) = \frac{jd}{\delta}$.

Note that $\ker(\eta)$ is generated (as a vector basis) by the ω_j for $1 \leq j \leq r-1$. Indeed, the few lines above show that the ω_j lie in the kernel of η and the first reduction formula shows that they are linearly independent. Since there are $\delta - 1$ of them, which is the dimension of $\ker(\eta)$, it follows that they form a basis of $\ker(\eta)$.

Now we determine the action of Frobenius on $\ker(\eta)$. Recall that $\mathcal{F}$ denotes the p-th power Frobenius, so $\mathcal{F}^n$ is the q-th power Frobenius.

Let $\omega_j = d\left(-\frac{r}{j}\frac{B_j}{y^j}\right)$ be one of the generators of $\ker(\eta)$. The Frobenius commutes with the operator d, so $\ker(\eta)$ is stable under the action of the Frobenius. This means that

$$\mathcal{F}^n(\omega_j) = \lambda_j \omega_l - r dQ, \quad \text{where } \ell = jp \bmod r \text{ and } Q = \sum_{k\geq 2} Q_k \frac{\tau^{k-1}}{y^\ell},$$

so

$$\frac{1}{j} d\left(\mathcal{F}^n\left(\frac{B_j}{y^j}\right)\right) = \frac{\lambda_j}{\ell} d\left(\frac{B_l}{y^\ell}\right) + dQ.$$

This implies

$$\frac{1}{j}\mathcal{F}^n\left(\frac{B_j}{y^j}\right) = \frac{\lambda_j}{\ell}\frac{B_l}{y^\ell} + \left(\frac{Q_2}{y^{r+\ell}} + \frac{Q_3}{y^{2r+\ell}} + \cdots\right).$$

We compute $\mathcal{F}\left(\frac{B_j}{y^j}\right) = B_j(x^p)\mathcal{F}(y)^{-j}$: as $B_j(x^p) = x^{b_j p} + \cdots$, we have (after normalization)

$$B_j(x^p) = u_a(x) y^{ar} + \cdots + u_0(x),$$

where $a = \frac{jp-\ell}{r} + \lfloor \frac{\ell}{d} \rfloor$ and u_a is monic of degree $\frac{d}{\delta} \times (\ell \bmod d)$ and $\deg(u_i) < d$, for $0 \leq i \leq a$. Indeed, $\deg(B_j(x^p)) = \frac{jpd}{\delta} = \frac{(ar+\ell)d}{\delta} = d\left(ar + \frac{\ell}{\delta}\right)$ so this gives the normalization above. Thus,

$$\begin{aligned}\mathcal{F}\left(\frac{B_j}{y^j}\right) &= (u_a(x)y^{ar} + \cdots + u_0(x))\left(y^{-jp}(1 + \frac{v_1}{y^r} + \frac{v_2}{y^{2r}} + \cdots)\right) \\ &= \frac{u_a(x)}{y^\ell} + \left(\frac{a_1}{y^{r+\ell}} + \frac{a_2}{y^{2r+\ell}} + \cdots\right),\end{aligned}$$

with u_a of degree $\frac{\ell d}{\delta}$ and $\ell = jp \bmod r$.

Proposition 7.2 shows that Frobenius applied to $\left(\frac{a_1}{y^{r+\ell}} + \frac{a_2}{y^{2r+\ell}} + \cdots\right)$ has the same form: that is, it has no term in $y^{-\ell}$. This means that

$$\mathcal{F}^n\left(\frac{B_j}{y^j}\right) = \frac{B_m(x)}{y^m} + \left(\frac{Q_1}{y^{r+m}} + \frac{Q_2}{y^{2r+m}} + \cdots\right),$$

where $m = jq \bmod r$ and B_m is monic of degree $\frac{md}{\delta}$, so $\lambda_j = \frac{m}{j}$ with $m = jq \bmod r$.

Let ϕ be the permutation of $\{\frac{r}{\delta}, 2\cdot\frac{r}{\delta}, \cdots, (\delta-1)\cdot\frac{r}{\delta}\}$ defined by

$$\phi : j \longmapsto jq \bmod r$$

(this is well defined because it acts on a set isomorphic to the subgroup of elements of order dividing δ in $\mathbb{F}_r^*$ and this is a permutation because q is prime to r). We have

$$\mathcal{F}^n(\omega_j) = \frac{\phi(j)}{j}\omega_{\phi(j)},$$

so the matrix of Frobenius with respect to our basis $W = (\omega_1, \cdots, \omega_{\delta-1})$ of the kernel of η has the form DP with D diagonal and P a permutation matrix. □

The previous proposition allows us to describe the extra factor appearing in the characteristic polynomial of the q-th power Frobenius acting on B'.

THEOREM 7.5. *The Weil polynomial P of $\mathcal{C}$ is*

$$P(t) = \frac{\chi_M(t)}{U(t)},$$

where $\chi_M(t)$ is the characteristic polynomial of the matrix corresponding to the q-th power Frobenius with respect to B' and

$$U(t) = \prod_{i|\delta\ ,\ i>1} (t^{k_i} - 1)^{\frac{\varphi(i)}{k_i}},$$

where k_i is the order of q in $\mathbb{Z}/\varphi(i)\mathbb{Z}$ and φ is the Euler totient function.

PROOF. Proposition 7.4 tells us that B' decomposes into a direct sum of two spaces including $H^1_{MW}(\mathcal{C}, \mathbb{Q}_q)$. This means that if we compute the characteristic polynomial of the Frobenius acting on B', then we have to remove an extra factor from it to recover the Weil polynomial. This extra factor $U(t)$ is the characteristic polynomial of the Frobenius acting on $\ker(\eta)$, whose action is described by Proposition 7.4. Since we can change the order of the vectors in the basis W of $\ker(\eta)$ (see the Proof of Proposition 7.4) without changing the characteristic polynomial, U is the characteristic polynomial of $\widetilde{D}\widetilde{P}$ where $\widetilde{P}$ is a block diagonal matrix of permutation (the blocks correspond to the disjoint cycles of the permutation ϕ acting on $\ker(\eta)$) , so $\chi_{\widetilde{D}\widetilde{P}}$ is a product of polynomials of the form

$$(t^k - \Delta_k),$$

where Δ_k is the determinant of the corresponding block of $\widetilde{D}\widetilde{P}$ and has the form

$$\Delta_k = \frac{\phi(j)}{j} \times \frac{\phi^2(j)}{\phi(j)} \times \cdots \times \frac{\phi^k(j)}{\phi^{k-1}(j)}.$$

As $\phi^k(j) = j$, the previous product is telescopic so $\Delta_k = 1$.

In order to describe the number and the lengths of these cycles, we show that the action of ϕ on $\ker(\eta)$ is equivalent to the action of the q-th power Frobenius on the $\delta - 1$ points at infinity

$$P_{k,\infty} = [1 : \zeta_r^k : 0], \quad \text{for } 1 \leq k \leq \delta - 1.$$

The map

$$\psi : \left\{ P_{k,\infty} = [1 : \zeta_r^k : 0] \in \mathcal{C}(\overline{\mathbb{F}_q}) : k \in [1, \delta-1] \right\} \longrightarrow \left\{ j = k\frac{r}{\delta} \in \mathbb{F}_r^\times \ : \ k \in [1, \delta-1] \right\}$$

defined by

$$\psi : P_{k,\infty} \longmapsto j = k\frac{r}{\delta}$$

is a bijection compatible with the action of the q-th power Frobenius and the action of ϕ. Indeed,

$$\begin{aligned}\psi\left(\Sigma^n(P_{k,\infty})\right) &= \psi([1:\zeta_r^{kq \bmod r}:0]) \\ &= \psi([1:\zeta_r^{kq \bmod \delta}:0]) \text{ because of the relations in } \mathbb{P}\left(\frac{r}{\delta},\frac{d}{\delta},1\right) \\ &= (kq \bmod \delta) \times \frac{r}{\delta} \\ &= (kq \times \frac{r}{\delta}) \bmod r \text{ because if } kq = a\delta + b \text{ then } kq\frac{r}{\delta} = ar + b\frac{r}{\delta} \\ &= \phi(k).\end{aligned}$$

This shows that U is the characteristic polynomial of the q-th power Frobenius acting on the $\delta - 1$ points at infinity of $\mathcal{C}$ described above.

In $\mathbb{P}(\frac{r}{\delta},\frac{d}{\delta},1)$, the points $[X:Y:Z]$ at infinity are those with $Z = 0$ and $Y^r = X^d$, which is equivalent to $\widetilde{Y}^\delta = \widetilde{X}^\delta$, with $\widetilde{Y} = Y^{\frac{r}{\delta}}$ and $\widetilde{X} = X^{\frac{d}{\delta}}$: that is, $T^\delta = 1$ where $T = \frac{\widetilde{Y}}{\widetilde{X}}$.

As we have

$$T^\delta - 1 \;=\; \prod_{i|\delta} \Phi_i(T) \;=\; (T-1) \prod_{i|\delta\ ,\ i>1} \Phi_i(T),$$

where Φ_i is the i-th cyclotomic polynomial, the Frobenius acts on the $\delta - 1$ points at infinity $P_{\infty,k}$ for $1 \le k < \delta$ as it acts on the roots of

$$\prod_{i|\delta, i>1} \Phi_i(T).$$

If i is an integer dividing δ, then the Frobenius acts on the $\varphi(i)$ roots of $\Phi_i(T)$ as a product of $\frac{\varphi(i)}{k_i}$ cycles of length k_i, where k_i is the order of q in $\mathbb{Z}/\varphi(i)\mathbb{Z}$. So the Frobenius map acts on these $\delta - 1$ points at infinity of $\mathcal{C}$ as a permutation whose corresponding permutation matrix has characteristic polynomial

$$\prod_{i|\delta, i>1} (t^{k_i} - 1)^{\frac{\varphi(i)}{k_i}}.$$

□

8. Numerical experiments

In this section, we give some experimental results obtained with a prototypical (and unoptimized) implementation of our algorithm in Magma 2.18. The experiments were run on a single core of a Xeon E5520 machine (2.26GHz, 72GB RAM). We tested our results by taking a random divisor on the curve, multiplying it by the supposed order of the Jacobian of the curve (which is the Weil polynomial evaluated at 1), and checking whether the resulting divisor is principal.

Example 8.1. We consider the genus 13 curve

$$\begin{aligned}\mathcal{C}_{3,15} : y^3 &= x^{15} + (2\alpha+5)x^{13} + 2\alpha x^{12} + \alpha x^{11} + (3\alpha+6)x^{10} \\ &\quad + 3x^9 + (2\alpha+4)x^8 + 4\alpha x^7 + 6\alpha x^6 + 6x^4 \\ &\quad + \alpha x^3 + (4\alpha+5)x^2 + (6\alpha+5)x\end{aligned}$$

defined over $\mathbb{F}_{49} = \mathbb{F}_7[\alpha]/\langle\alpha^2 - \alpha + 4\rangle$. After 585 seconds, our implementation returns the Weil polynomial of $\mathcal{C}$, whose Weil coefficients are

$$a_1 = 4, \quad a_2 = -88, \quad a_3 = -317, \quad a_4 = 3477, \quad a_5 = 45743, \quad a_6 = -38408,$$
$$a_7 = -3064081, \quad a_8 = 1826186, \quad a_9 = 105964107, \quad a_{10} = 178170657,$$

$$a_{11} = -3878128722, \quad a_{12} = -10860792624 \quad \text{and } a_{13} = 227741125446.$$

EXAMPLE 8.2. We consider the genus 26 curve

$$\begin{aligned}\mathcal{C}_{5,15} : y^5 &= x^{15} + (4\alpha+7)x^{13} + (4\alpha+6)x^{12} + (2\alpha+4)x^{11} \\ &+ (10\alpha+4)x^{10} + (\alpha+10)x^9 + 4x^8 + 2x^7 + 6x^6 \\ &+ (3\alpha+1)x^5 + 10x^4 + (10\alpha+1)x^3 + (5\alpha+9)x^2 \\ &+ (7\alpha+4)x + 2\alpha + 6\end{aligned}$$

defined over $\mathbb{F}_{121} = \mathbb{F}_{11}[\alpha]/\langle \alpha^2 - \alpha + 4\rangle$. After 23922 seconds, our implementation returns the Weil polynomial of $\mathcal{C}$ whose Weil coefficients are

$$a_1 = 36,\ a_2 = 418,\ a_3 = 3928,\ a_4 = 107603,\ a_5 = 1546802,\ a_6 = 10195080,$$
$$a_7 = 189193348,\ a_8 = 3908194517,\ a_9 = 35529836037,\ a_{10} = 323855056565,$$
$$a_{11} = 6026279205222,\ a_{12} = 71054667707163,\ a_{13} = 577639402235514,$$
$$a_{14} = 7788857330417489,\ a_{15} = 103362684561282136,$$
$$a_{16} = 988282517113615745,\ a_{17} = 11354454883387292669,$$
$$_{18} = 122508522344304060111,\ a_{19} = 999211815604433952646,$$
$$a_{20} = 13694995222065645049886,\ a_{21} = 174130364097714846506217$$
$$a_{22} = 1066845743104788110404502,\ _{23} = 11897270459284483568657805,$$
$$a_{24} = 24375922693990238345952627 5,\ a_{25} = 19251288794802012387593080 35,$$
$$\text{and } a_{26} = 81302846530212152396447907725.$$

EXAMPLE 8.3. We consider the genus 45 curve

$$\mathcal{C}_{11,11} : y^{11} = x^{11} + 21x^9 + 22x^8 + 12x^7 + 14x^6 + 5x^4 + 15x^3 + 6x^2 + 15x + 11$$

defined over $\mathbb{F}_{23}$. After 159200 seconds, our implementation returns the Weil polynomial of $\mathcal{C}$ whose Weil coefficients are:

$$a_1 = -10,\ a_2 = 148,\ a_3 = -1172,\ a_4 = 11400,\ a_5 = -75082,\ a_6 = 583607,$$
$$a_7 = -3423792,\ a_8 = 23458758,\ a_9 = -127681770,\ a_{10} = 815749654,$$
$$a_{11} = -4274768142,\ a_{12} = 26177112830,\ a_{13} = -133290333147,$$
$$a_{14} = 792181088309,\ a_{15} = -3931625501060,\ a_{16} = 22819266210165,$$
$$a_{17} = -110481821962459,\ a_{18} = 633740960651940,\ a_{19} = -3001343844798677,$$
$$a_{20} = 17054767132345719,\ a_{21} = -79052006236498542,$$
$$a_{22} = 445634829426753123,\ a_{23} = -2018975937263556165,$$
$$a_{24} = 24375922693990238345952627 5,\ a_{25} = -50378603603766216893,$$
$$a_{26} = 281146158641010525301,\ a_{27} = -1239849286459249269112,$$
$$a_{28} = 6921368868854435563991,\ a_{29} = -3028723789994138911185 0,$$
$$a_{30} = 1687194246872522640767 67,\ a_{31} = -7285843030247638250038 60,$$
$$a_{32} = 4051750456875540838493246,\ a_{33} = -1720766556504792178335341 4,$$
$$a_{34} = 9553153794464572098080333 4,\ a_{35} = -3985150326246674041871542 80,$$
$$a_{36} = 2220486855862732905431832556,\ a_{37} = -9115467662197167357206988 72,$$
$$a_{38} = 5098757240007702925025305848 3,$$
$$a_{39} = -2072635069308839338584039222 80,$$
$$a_{40} = 1165874930218286023405099204 275,$$
$$a_{41} = -4712376446054941126784485443 520,$$
$$a_{42} = 2663176150610149625881689927 4283,$$
$$a_{43} = -1077665343462102346861122820 45945,$$

$$a_{44} = 610647567000069960495606605432680,$$
$$\text{and } a_{45} = -247240714379333501838939433648611 1.$$

EXAMPLE 8.4. We consider the genus 57 curve

$$\begin{aligned}\mathcal{C}_{7,21} : y^7 &= x^{21} + \alpha^{166}x^{19} + \alpha^{12}x^{18} + \alpha^{64}x^{17} + \alpha^{102}x^{16} + \alpha^{166}x^{15} + 12x^{14}\\ &+ \alpha^{25}x^{13} + \alpha^{68}x^{11} + \alpha^{117}x^{10} + \alpha^{8}x^{9} + \alpha^{15}x^{8} + \alpha^{16}x^{7} + \alpha^{127}x^{6}\\ &+ \alpha^{90}x^{5} + \alpha^{43}x^{4} + \alpha^{128}x^{3} + \alpha^{40}x^{2} + \alpha^{125}x + \alpha^{99}\end{aligned}$$

defined over $\mathbb{F}_{169} = \mathbb{F}_{13}[\alpha]/\langle\alpha^2 - \alpha + 2\rangle$. After 380881 seconds, our implementation returns the Weil polynomial of $\mathcal{C}$, which factors as

$$P_{\mathcal{C}_{7,21}}(T) = (T + 13)^6 P(T)^2,$$

where P is the Weil polynomial of a 27-dimensional abelian variety whose Weil coefficients are:

$$a_1 = 14,\ a_2 = 224,\ a_3 = 3804,\ a_4 = 68075,\ a_5 = 650370,\ a_6 = 8859458,$$
$$a_7 = 72307214,\ a_8 = 1083567163,\ a_9 = -157139189,\ a_{10} = -20620569697,$$
$$a_{11} = -2357957261121,\ a_{12} = -16670241272334,\ a_{13} = -448263291116144,$$
$$a_{14} = -145927900246555,\ a_{15} = -23388115214168173,$$
$$a_{16} = 647629219619169060,\ a_{17} = 6886478186860664665,$$
$$a_{18} = 328920785102728658021,\ a_{19} = 2370798532171844617115,$$
$$a_{20} = 52751582248734601974196,\ a_{21} = 3267492521270936392530802,$$
$$a_{22} = 56417623169758856819644 74,\ a_{23} = -35674382266353914319048358,$$
$$a_{24} = -32158762836019054780253774 0,\ a_{25} = -19029290083673947265278225863,$$
$$a_{26} = -15208490489425108105544 3498722,$$
$$\text{and } a_{27} = -39833837478395886806453200443 53.$$

References

[BEd13] Amnon Besser, François-Renaud Escriva, and Rob de Jeu, *Frobenius lifts and point counting for smooth curves*, Preprint available at `http://arxiv.org/abs/1306.5102`, June 2013.

[BGS07] Alin Bostan, Pierrick Gaudry, and Éric Schost, *Linear recurrences with polynomial coefficients and application to integer factorization and Cartier-Manin operator*, SIAM J. Comput. **36** (2007), no. 6, 1777–1806. MR 2299425 (2008a:11156)

[CDV06] Wouter Castryck, Jan Denef, and Frederik Vercauteren, *Computing zeta functions of nondegenerate curves*, IMRP Int. Math. Res. Pap. (2006), Art. ID 72017, 57. MR 2268492 (2007h:14026)

[CHV08] Wouter Castryck, Hendrik Hubrechts, and Frederik Vercauteren, *Computing Zeta Functions in families of $C_{a,b}$ curves using deformation*, Algorithmic Number Theory (AlfredJ. Poorten and Andreas Stein, eds.), Lecture Notes in Comput. Sci., vol. 5011, Springer Berlin Heidelberg, 2008, pp. 296–311. MR 2467856 (2010d:11148)

[DV06] Jan Denef and Frederik Vercauteren, *Counting points on C_{ab} curves using Monsky-Washnitzer cohomology*, Finite Fields Appl. **12** (2006), no. 1, 78–102. MR 2190188 (2007c:11075)

[Edi03] Bas Edixhoven, *Point counting after Kedlaya. EIDMA-Stieltjes Graduate course*, 2003, Available at `http://pub.math.leidenuniv.nl/~edixhovensj/oww/mathofcrypt/carls_edixhoven/kedlaya.pdf`.

[GG01] Pierrick Gaudry and Nicolas Gürel, *An extension of Kedlaya's point-counting algorithm to superelliptic curves*, Advances in cryptology—ASIACRYPT 2001 (Gold Coast), Lecture Notes in Comput. Sci., vol. 2248, Springer, Berlin, 2001, pp. 480–494. MR 1934859 (2003h:11159)

[GG03] ———, *Counting points in medium characteristic using Kedlaya's algorithm*, Experiment. Math. **12** (2003), no. 4, 395–402. MR 2043990 (2005b:11084)

[GH00] Pierrick Gaudry and Robert Harley, *Counting points on hyperelliptic curves over finite fields*, Algorithmic number theory (Leiden, 2000), Lecture Notes in Comput. Sci., vol. 1838, Springer, Berlin, 2000, pp. 313–332. MR 1850614 (2002f:11072)

[GKS11] Pierrick Gaudry, David Kohel, and Benjamin Smith, *Counting points on genus 2 curves with real multiplication*, Advances in cryptology—ASIACRYPT 2011, Lecture Notes in Comput. Sci., vol. 7073, Springer, Heidelberg, 2011, pp. 504–519. MR 2935020

[GS12] Pierrick Gaudry and Éric Schost, *Genus 2 point counting over prime fields*, J. Symbolic Comput. **47** (2012), no. 4, 368–400. MR 2890878

[Har07] David Harvey, *Kedlaya's algorithm in larger characteristic*, Int. Math. Res. Not. IMRN (2007), no. 22, Art. ID rnm095, 29. MR 2376210 (2009d:11096)

[Har12] Michael C. Harrison, *An extension of Kedlaya's algorithm for hyperelliptic curves*, J. Symbolic Comput. **47** (2012), no. 1, 89–101. MR 2854849

[Ked01] Kiran S. Kedlaya, *Counting points on hyperelliptic curves using Monsky–Washnitzer cohomology*, J. Ramanujan Math. Soc. **16** (2001), no. 4, 323–338. MR 1877805 (2002m:14019)

[Mes00] Jean-François Mestre, *Lettre à Gaudry et Harley*, December 2000, Available at `http://www.math.jussieu.fr/~mestre/lettreGaudryHarley.ps`.

[Mes02] ———, *Notes of a talk given at the seminar of cryptography of Rennes*, 2002, Available at `http://www.math.jussieu.fr/~mestre/rennescrypto.ps`.

[Min10] Moritz Minzlaff, *Computing zeta functions of superelliptic curves in larger characteristic*, Math. Comput. Sci. **3** (2010), no. 2, 209–224. MR 2608297 (2011i:11094)

[Pil88] Jonathan Pila, *Frobenius maps of Abelian varieties and finding roots of unity in finite fields*, Ph.D. thesis, 1988, Thesis (Ph.D.)–Stanford University, p. 56. MR 2637049

[Pil90] ———, *Frobenius maps of Abelian varieties and finding roots of unity in finite fields*, Math. Comp. **55** (1990), no. 192, 745–763. MR 1035941 (91a:11071)

[PT13] Sebastian Pancratz and Jan Tuitman, *Improvements to the deformation method for counting points on smooth projective hypersurfaces*, Preprint available at `http://arxiv.org/abs/1307.1250`, July 2013.

[Rei02] Miles Reid, *Graded rings and varieties in weighted projective space*, 2002, Available at `http://www.warwick.ac.uk/~masda/surf/more/grad.pdf`.

[Sat00] Takakazu Satoh, *The canonical lift of an ordinary elliptic curve over a finite field and its point counting*, J. Ramanujan Math. Soc. **15** (2000), no. 4, 247–270. MR 1801221 (2001j:11049)

[Sch85] René Schoof, *Elliptic curves over finite fields and the computation of square roots mod p*, Math. Comp. **44** (1985), no. 170, 483–494. MR 777280 (86e:11122)

[Sch95] ———, *Counting points on elliptic curves over finite fields*, J. Théor. Nombres Bordeaux **7** (1995), no. 1, 219–254, Les Dix-huitièmes Journées Arithmétiques (Bordeaux, 1993). MR 1413578 (97i:11070)

[Tui14] Jan Tuitman, *Counting points on curves using a map to* $\mathbb{P}^1$, Preprint available at `http://arxiv.org/abs/1402.6758`, February 2014.

[vdP86] Marius van der Put, *The cohomology of Monsky and Washnitzer*, Mém. Soc. Math. France (N.S.) (1986), no. 23, 4, 33–59, Introductions aux cohomologies p-adiques (Luminy, 1984). MR 865811 (88a:14022)

[vzGG03] Joachim von zur Gathen and Jürgen Gerhard, *Modern computer algebra*, second ed., Cambridge University Press, Cambridge, 2003. MR 2001757 (2004g:68202)

Laboratoire d'Informatique de l'École polytechnique, 1 rue Honoré d'Estienne d'Orves, Bâtiment Alan Turing, Campus de l'École Polytechnique, 91120 Palaiseau, France

E-mail address: `goncalves@lix.polytechnique.fr`