An Extension of Kedlaya's Point-Counting Algorithm to Superelliptic Curves

Pierrick Gaudry and Nicolas Gürel

LIX, École polytechnique, 91128 Palaiseau Cedex, France
{gaudry, gurel}@lix.polytechnique.fr

Abstract. We present an algorithm for counting points on superelliptic curves $y^r = f(x)$ over a finite field $\mathbb{F}_q$ of small characteristic different from r. This is an extension of an algorithm for hyperelliptic curves due to Kedlaya. In this extension, the complexity, assuming r and the genus are fixed, is $O(\log^{3+\varepsilon} q)$ in time and space, just like for hyperelliptic curves. We give some numerical examples obtained with our first implementation, thus proving that cryptographic sizes are now reachable.

1 Introduction

In the past few years a lot of candidates have been proposed to enlarge the set of groups that can be used in protocols based on the discrete logarithm problem like Diffie–Hellman or ElGamal. Beside the classical multiplicative groups of finite fields, the most famous are certainly the systems based on elliptic curves [21, 26]. Indeed, for these systems the only general attacks known are variants of the Pollard Rho method which require exponential time computation; in practice it means that the key size is much shorter than in a system that uses finite fields. Thereafter, systems based on hyperelliptic curves were proposed [22]. They seem to have the same advantages as elliptic curve cryptosystems (at least when the genus is less than 4 [1, 14]).

More recently, systems based on the discrete logarithm problem in the Jacobians of other curves were designed. Namely, in the literature, we can now find algorithms for working in Jacobians of superelliptic curves [13] and of C_{ab} curves [2]. Several works related to these curves have already been published, concerning security issues [4], efficiency [17, 6], building curves with known number of points [3], or possible use in a Weil restriction attack on elliptic curves [5]. The next step for studying the possible cryptographic use of these curves is to conceive an algorithm for counting points of the Jacobian of a *random* curve. Indeed, this is thought to be one of the most secure ways of building a cryptosystem by a large part of the community.

In the case of elliptic curves, this problem of point counting has been a challenge of the past 15 years and nowadays we have satisfactory solutions. When the characteristic of the base field is large the best known method is Schoof's algorithm and all the improvements leading to the so-called Schoof–Elkies–Atkin algorithm. We refer to [7] or [23] for surveys of these techniques and to the references therein. Besides some theoretical results [29] and an attempt to make them practical [15], extending the SEA algorithm to higher genus has not yet proven to be enough for cryptographic sizes. The situation is quite different in small characteristic: two years ago, Satoh [32] showed that p-adic methods using the canonical lift could lead to an algorithm asymptotically faster than SEA. Some work has been done consequently on the subject to extend it to characteristic 2 [33, 9], to implement it and obtain new records [9], to use less memory [34], and to combine it with an early-abort strategy for generating secure curves [10]. Mestre, Harley and Gaudry recently proposed a related algorithm, based on the arithmetic-geometric mean, for elliptic curves and hyperelliptic curves of genus 2 in characterstic 2; a nice feature of this technique is that it does not explicitly make use of j-invariants, of modular equations nor of Vélu-type formulae, and these had previously been the main obstructions to generalizing beyond the elliptic case. However, the AGM method does not seem to extend easily to non-hyperelliptic curves. Another approach, also using p-adic methods but not based on canonical lifting, has been proposed by Kedlaya [18]. His method applies to hyperelliptic curves in small odd characteristic. The complexity in time is $O(\log^{3+\varepsilon} q)$, for curves over $\mathbb{F}_q$ of fixed genus, i.e. the same as all the variants of Satoh's method and the complexity in space is $O(\log^3 q)$ which is the same as Satoh's original algorithm, but bad compared to the algorithm of [34] or AGM.

The contribution of this paper is twofold: firstly we show that Kedlaya's algorithm can be extended in a rather straightforward way to superelliptic curves; secondly we report some results obtained with our first implementation written in Magma. To our knowledge, these are the first published point counting computations for random hyperelliptic and superelliptic curves of cryptographic sizes.

The paper is organized as follows: after recalling some basics about curves and p-adic numbers, we describe Kedlaya's original algorithm and show how to adapt it for superelliptic curves. Then we give some more details on the way these algorithms can be handled in practice and we estimate the complexity. We conclude by numerical examples and remarks about the use of these curves in cryptography.

2 Background on Algebraic Curves and p-adic Number Rings

In this section, we recall some basic facts about algebraic curves over finite fields and p-adic numbers. We shall not give precise definitions and we refer the reader to classical books on the subject ([12, 20, 19, 24] for instance).

2.1 Hyperelliptic and Superelliptic Curves

Let $\mathbb{F}_q$ be a finite field with $q = p^n$ elements. We shall consider only two types of curves over $\mathbb{F}_q$, namely hyperelliptic and superelliptic curves.

Definition 1 *A* superelliptic *curve is a plane curve $\mathcal{C}$ which admits an affine equation of the form*

$$y^r = f(x),$$

where r is a prime different from p and f is monic, squarefree of degree d coprime to r.

With such a definition, $\mathcal{C}$ is non-singular in its affine part, and admits a unique place of degree 1 at infinity. Moreover its genus is given by $g = \frac{(d-1)(r-1)}{2}$.

Definition 2 *In characteristic different from 2, a* hyperelliptic *curve is a superelliptic curve whose equation is of the form $y^2 = f(x)$, with $r = 2$ and f of degree $2g+1$.*

Note that there exists a more general definition of hyperelliptic curves which do not exclude the case of characteristic 2. But the algorithms we will describe work only for this particular case.

Let $\mathcal{C}$ be a superelliptic curve of genus g. Associated to this curve, one can define its *Jacobian*, noted $\mathrm{J}(\mathcal{C})$, which is a finite abelian group. In the past few years, several algorithms were developed to compute explicitly in this group [13, 2, 17, 6]. The next step is to study the order of $\mathrm{J}(\mathcal{C})$. For this the q-th power Frobenius endomorphism and its characteristic polynomial $\chi(T)$ are key tools. More precisely, $\chi(T)$ can be written as

$$\chi(T) = \sum_{i=0}^{2g} a_i T^i, \quad \text{with} \quad a_{2g} = 1, \quad a_i = q^{g-i} a_{2g-i} \quad \text{for } i = 0, \ldots, g-1,$$

and all its roots have absolute value $\sqrt{q}$. This is essentially the Riemann Hypothesis for zeta functions of curves. For us, the interesting fact is that $\#\mathrm{J}(\mathcal{C}) = \chi(1)$. Our goal in this paper is to compute $\chi(T)$ and to obtain $\#\mathrm{J}(\mathcal{C})$ as a byproduct.

2.2 The Ring $\mathbb{Z}_q$

Let K be the (unique up to isomorphism) unramified extension of degree n of $\mathbb{Q}_p$; its residual field is $\mathbb{F}_q$. We denote by $\mathbb{Z}_q$ the ring of integers of K. In order to construct it, we can start with the polynomial $\overline{P}(t)$ which defines $\mathbb{F}_q$ as an algebraic extension of $\mathbb{F}_p$; we then consider the extension

$$\mathbb{Z}_q := \mathbb{Z}_p[t]/(P(t)),$$

where the polynomial $P(t)$ is obtained from $\overline{P}(t)$ by lifting trivially its coefficients to p-adic integers. In practice, an element z of $\mathbb{Z}_q$ can be represented as a polynomial $z = z_{n-1}t^{n-1} + z_{n-2}t^{n-2} + \cdots + z_1 t + z_0$ taken modulo $P(t)$ and where the z_i are integers modulo a power of p called the *precision* at which the computation is done.

It can be shown that the Galois group of K over $\mathbb{Q}_p$ is cyclic. We will denote by σ the unique generator, also called Frobenius, of this Galois group that reduces modulo p to the p-th power Frobenius in $\mathbb{F}_q$. There is no trivial formula for writing z^σ for an element z in $\mathbb{Z}_q$ expressed on a polynomial basis as above. Later on, we will describe how to precompute t^σ and then z^σ is obtained as follows:

$$z^\sigma = \left(\sum_{i=0}^{n-1} z_i t^i\right)^\sigma = \sum_{i=0}^{n-1} z_i (t^\sigma)^i.$$

3 Kedlaya's Algorithm and its Extension

3.1 Overview of Kedlaya's Algorithm for Hyperelliptic Curves

Let $\mathcal{C}$ be a hyperelliptic curve of genus g given by its equation $y^2 = \overline{f}(x)$ over $\mathbb{F}_q$. Following the construction of Kedlaya (see also [20], page 72), we consider the curve $\mathcal{C}'$ obtained from $\mathcal{C}$ by removing the point at infinity and the points with vertical tangent (i.e. $y = 0$).

There is a way to lift the coordinate ring of $\mathcal{C}'$ called the weak completion [27], with the nice property that its cohomology verifies a “Lefschetz trace formula” [28] and hence gives information about the cardinalities of the initial curve $\mathcal{C}$.

Taking a lowbrow point of view in which we can forget about the curve $\mathcal{C}'$, we shall work on the vector space generated over the p-adic number field K by the following differential forms:

$$\mathcal{D} = \left\langle \frac{x^i dx}{y}; \quad i \in [0, 2g-1] \right\rangle,$$

in which we have the relations coming from the equation of the curve and $d\varphi(x, y) \equiv 0$ for every rational function φ. On the differential forms one can define a Frobenius action which is compatible with the p-th power Frobenius on $\mathcal{C}$: take $x^\sigma = x^p$, y^σ given by $(y^\sigma)^2 = f(x)^\sigma$ and $(dx)^\sigma = px^{p-1}dx$. Kedlaya shows in a constructive way that the space $\mathcal{D}$ is stable under the action of this σ. Hence σ is an endomorphism of a vector space of dimension $2g$; and everything is done in order for its characteristic polynomial to be closely related to the $\chi(T)$ we are looking for. The heart of Kedlaya's algorithm is then to compute the matrix of σ for the given basis of $\mathcal{D}$.

For each i in $[0, 2g-1]$,

$$\left(\frac{x^i dx}{y}\right)^\sigma = \frac{1}{y^\sigma} p x^{ip+p-1} dx,$$

therefore the tricky part is the computation of $\frac{1}{y^\sigma}$. This is not defined in a lifted coordinate ring because it involves a square root and that is a reason why we use the weak completion. From a practical point of view, it means that we shall be able to expand $\frac{1}{y^\sigma}$ as a power series in $\tau = \frac{1}{y^2}$: starting with the definition

$(y^\sigma)^2 = f(x)^\sigma$, we have

$$\begin{aligned}\tfrac{1}{y^\sigma} &= \left(f(x)^\sigma\right)^{-1/2}\\ &= \left(f(x)^\sigma - f(x)^p + f(x)^p\right)^{-1/2}\\ &= \left(f(x)^p\right)^{-1/2}\left(1+\tfrac{f(x)^\sigma - f(x)^p}{f(x)^p}\right)^{-1/2}\\ &= \tfrac{1}{y^p}\left(1+\tau^p(f(x)^\sigma - f(x)^p)\right)^{-1/2} \ .\end{aligned}$$

By the usual power series expansion of $(1+X)^{-1/2}$ we get an expression of the form

$$\frac{1}{y^\sigma} = y^{-p}\sum_{k\geq 0} P_k(x)\tau^{pk} = y^{-1}\tau^{(p-1)/2}\sum_{k\geq 0} P_k(x)\tau^{pk}.$$

Note that p divides $(f(x)^\sigma - f(x)^p)$ so that the power of p dividing $P_k(x)$ tends to infinity as k grows (actually this is what is expected due to the theoretical construction of the weak completion). We can now write

$$\left(\frac{x^i dx}{y}\right)^\sigma = \left(\sum_{k\geq 0} Q_k(x)\tau^k\right)\frac{dx}{y},$$

where $Q_k(x)$ are polynomials. The algorithm proceeds as follows: we compute this expression up to some precision in τ, and then we use the relations in $\mathcal{D}$ described above to reduce the expression to a polynomial of degree at most $2g-1$, times $\frac{dx}{y}$. In this way we shall prove that $\mathcal{D}$ is indeed σ-stable and moreover we obtain an explict description of the action of σ on the basis. For this we will use three strategies of reduction:

Red 1. First of all, using the equation of the curve, one can write

$$Q_k(x)\tau^k = (\alpha_k(x)f(x) + \beta_k(x))\tau^k = \alpha_k(x)\tau^{k-1} + \beta_k(x)\tau^k,$$

where α_k and β_k are the quotient and the remainder in the division of Q_k by f. Therefore one can assume that $Q_k(x)$ is of degree at most $2g$ for all k, except for $Q_0(x)$ for which one can show that the degree is at most $2pg-1$.

Red 2. Then we use the relations of cohomology to rewrite the series in the form $Q(x)\frac{dx}{y}$. Fix $k \geq 1$ and consider the term $Q_k(x)\tau^k\frac{dx}{y}$. Let $U(x)$ and $V(x)$ be such that $Q_k(x) = U(x)f(x) + V(x)f'(x)$ (they do exist because f is squarefree). Using

$$d\left(\frac{V(x)}{y^{2k-1}}\right) \equiv 0,$$

one obtains

$$Q_k(x)\,\tau^k\frac{dx}{y} \equiv \left(U(x) + \frac{2}{2k-1}V'(x)\right)\,\tau^{k-1}\frac{dx}{y}.$$

Repeating this for decreasing k's, we can rewrite everything on the constant term of the series.

Red 3. Finally, in the expression $Q(x)\frac{dx}{y}$ that we obtained, one can reduce the degree δ of Q to at most $2g-1$ in the following way. Assume $\delta \geq 2g$: using

$$d(x^{\delta-2g}y) \equiv 0,$$

one gets a polynomial of degree δ that can be subtracted from Q.

At this point, we have computed a $2g \times 2g$ matrix M such that

$$\begin{pmatrix} \frac{dx}{y} \\ \vdots \\ \frac{x^{2g-1}dx}{y} \end{pmatrix}^{\sigma} = M \begin{pmatrix} \frac{dx}{y} \\ \vdots \\ \frac{x^{2g-1}dx}{y} \end{pmatrix}.$$

Most of the operations done during the computation involve elements of $\mathbb{Z}_q$, but at the end we may have to divide by small powers of p. Finally the coefficients of M lie in $p^{-s}\mathbb{Z}_q$ with a small, predictable s, which depends only on p and g.

The final step is then to compute the characteristic polynomial of the matrix

$$MM^{\sigma}\cdots M^{\sigma^{n-1}},$$

which has coefficients in $\mathbb{Z}_2$ and is a p-adic approximation of $\chi(T)$.

3.2 Superelliptic Curves

Let $\mathcal{C}$ be a superelliptic curve given by its equation $y^r = \overline{f}(x)$ with $\overline{f}$ of degree d over $\mathbb{F}_q$. The theory is exactly the same as for hyperelliptic curves. In the present case, the space of differential forms we consider is

$$\left\langle \frac{x^i dx}{y^j};\quad i \in [0, d-2],\ j \in [1, r-1] \right\rangle.$$

The Frobenius action lifting the p-th power Frobenius on $\mathcal{C}$ is defined similarly: take $x^\sigma = x^p$, y^σ given by $(y^\sigma)^r = f(x)^\sigma$ and $(dx)^\sigma = px^{p-1}dx$.

Again, the space of differential forms has been chosen such that it is stable under the action of σ; we will now describe the reduction process which allows us to rewrite $\left(\frac{x^i dx}{y^j}\right)^\sigma$ over the basis. Fix an $i \in [0, d-2]$ and a $j \in [1, r-1]$. We can write $\left(\frac{1}{y^j}\right)^\sigma$ as a power series

$$\left(\frac{1}{y^j}\right)^\sigma = y^{-jp}\left(1 + \frac{f(x)^\sigma - f(x)^p}{y^{rp}}\right)^{-j/r} = y^{-jp}\sum_{k\geq 0} P_k(x)\tau^{pk},$$

where we have set $\tau = y^{-r}$. Hence we can write

$$\left(\frac{x^i dx}{y^j}\right)^\sigma = \left(\sum_{k\geq 0} Q_k(x)\tau^k\right)\frac{dx}{y^{jp \bmod r}}.$$

In the following, we let $\ell = jp \bmod r$. We now proceed with three reduction steps similar to those we had for hyperelliptic curves.

Red 1. First, use the equation of the curve to obtain a series where the $Q_k(x)$ are of degree at most $d-1$, except for the first one.

Red 2. Then, rewrite the term in τ^k as a term in τ^{k-1}. For $k \geq 1$, let $U(x)$ and $V(x)$ be such that $Q_k(x) = U(x)f(x) + V(x)f'(x)$, one has

$$Q_k(x)\,\tau^k \frac{dx}{y^\ell} \equiv \left(U(x) + \frac{r}{r(k-1)+\ell} V'(x)\right) \tau^{k-1} \frac{dx}{y^\ell}\ .$$

Red 3. Finally, we are left with an expression of the form $Q(x)\frac{dx}{y^\ell}$, where $Q(x)$ is a polynomial of degree δ that we can reduce to degree at most $d-2$: assume $\delta \geq d-1$, the exact differential $d(x^{\delta-d+1}y^{r-l}) \equiv 0$ gives a polynomial of degree δ that can be subtracted from $Q(x)$.

We obtain a $2g \times 2g$ matrix M and we conclude as before by taking the characteristic polynomial of its "norm".

Note that the differential forms in $\frac{dx}{y^j}$ are sent by σ to the subspace generated by forms in $\frac{dx}{y^\ell}$ with $\ell = jp \bmod r$. As a consequence, M is a matrix that can be viewed in blocks of size $d-1$, with the property that there is exactly one non-zero block in each row block and each column block.

4 Details and Complexity

4.1 Precision of the Computation

The intermediate result obtained from the algorithm of section 3 is an approximation of the polynomial $\chi(T)$ that we are looking for, and by computing to sufficient precision we can determine it exactly. Two parameters have to be tuned, to ensure that at the end we get enough information to conclude. The first is the p-adic precision p^ν at which we truncate elements of $\mathbb{Z}_p$. The second is the τ-adic precision at which we truncate the series.

Bounds on the coefficients of $\chi(T)$ can be deduced from the bounds on its roots: $|a_i| \leq \binom{2g}{i} q^{i/2}$ for $i \in [1, g]$. We assume that q is large compared to the genus, so that a_g determines the required precision. Hence we need to know $\chi(T)$ modulo $\lceil 2\binom{2g}{g} q^{g/2} \rceil$ to be sure to recover all the coefficients. Therefore the working precision should be at least

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

The precision in τ is more problematic: at first sight it is not clear that we do not need *all* the terms of the series to get a result which makes sense even modulo p. Actually in the power series expansion, one can see that the coefficient in τ^k (which is a polynomial over $\mathbb{Z}_q$) is divisible by a power of p which grows to infinity at the speed of k/p. Hence it appears that the precision μ in τ should be at least p times the p-adic precision ν. Moreover, the reduction process also perturbs things: starting with a term $Q_k(x)\tau^k\frac{dx}{y^\ell}$, with p^m dividing $Q_k(x)$, one reduces to a differential form $Q(x)\frac{dx}{y^\ell}$ and p^m does not divide $Q(x)$ any more. In Lemma 1 of [18], Kedlaya shows that we can bound the loss of precision by $\log_p(rk+\ell)$. Acordingly, it is sufficient to enlarge μ slightly to ensure that at the end we have the required precision. A tedious calculation leads to the following choice for μ: we take the smallest μ such that

$$\mu > p\nu - \frac{p}{r} + p\log_p((r+1)\mu - 1).$$

4.2 Detailed Algorithm

We summarize the algorithm in the following:

Input: A superelliptic curve $y^r = \overline{f}(x)$ over $\mathbb{F}_q$, $q = p^n$, the degree of f is noted d, $g = \frac{(d-1)(r-1)}{2}$.
Output: The characteristic polynomial $\chi(T)$.

1. Set the p-adic working precision $\nu = \lceil \log_p(2\binom{2g}{g}q^{g/2}) \rceil$ and set the maximal precision μ for the series to be the smallest value such that $\mu > p\nu - \frac{p}{r} + p\log_p((r+1)\mu - 1)$.
2. Let $S = 1 + (f(x)^\sigma - f(x)^p)\,\tau^p$, where $f(x)$ is the polynomial $\overline{f}(x)$ where the coefficients are lifted arbitrarily from $\mathbb{F}_q$ to $\mathbb{Z}_q$.
3. Compute $S^{-1/r}$ as a truncated series in τ, to precision τ^μ.
   For this, use a Newton iteration $X \leftarrow \frac{1}{r}((r+1)X - SX^{r+1})$, initialized with $X = 1$. At each step in the recursion, use **Red1** to keep the coefficients of the series of degree at most $d-1$.
4. Compute $S^{-j/r}$ for $j \in [2, r-1]$ up to precision τ^μ. This is done by multiplying $S^{-1/r}$ by itself repeatedly; again, use **Red1** after each multiplication.
5. For each $i \in [0, d-2]$ and $j \in [1, r-1]$ do
   a. Compute $\omega_{ij} = \left(\frac{x^i\,dx}{y^j}\right)^\sigma = p\tau^{jp \operatorname{div} r}x^{ip+p-1}S^{-j/r}\frac{dx}{y^{jp \bmod r}}$.
   b. Use **Red 2** to write ω_{ij} in the form $Q(x)\frac{dx}{y^{jp \bmod r}}$.
   During this reduction it is sometimes necessary to divide by an integer which is divisible by p. In theory, this ought to reduce the precision of the computation. Actually, when this occurs, one adds some arbitrary noise to "force" the precision to remain maximal. This strange way of doing things does not actually affect the final result because this noise will cancel out during the whole process. This is ensured by Lemma 1 in [18], which extends naturally to the superelliptic case.
   c. Use **Red 3** to reduce the degree of $Q(x)$.
6. Compute the matrix M, its norm and its characteristic polynomial $\widetilde{\chi}(T) = \sum_{0 \le k \le 2g} \tilde{a}_k T^k$.
7. For $k \in [1, g]$, find the integer a_k in $[-\frac{p^\nu}{2}, \frac{p^\nu}{2}]$ congruent to $\tilde{a}_k$ modulo p^ν. Return the corresponding $\chi(T)$.

4.3 Complexity

For the complexity analysis we shall make the following assumptions:

- The characteristic p is fixed;
- The parameters r and d of the curves are fixed, hence also the genus;
- Each time we have to do a multiplication between two elements of a rather complicated structure (truncated series over polynomials over polynomials over integers), we assume that we pack everything into large integers and that we use Schönhage's fast multiplication algorithm. A multiplication between two objects of bit-size N is then assumed to take time $O(N^{1+\varepsilon})$.

In Step 2 we have to apply Frobenius to some elements of $\mathbb{Z}_q$. For this, note that t being a root of $P(t)$, so is t^σ. Therefore, t^σ can be obtained by a Newton iteration $X \leftarrow X - P(X)/P'(X)$ initialized with t^p. This is just a precomputation and moreover the cost is comparable to the rest of the algorithm. Thereafter, it is possible to obtain the Frobenius of an element in $\mathbb{Z}_q$ in time $O(n^{3+\varepsilon})$.

Step 3 is a Newton lifting. The cost is bounded by a constant times the cost of the last iteration. This last iteration costs a few multiplications between objects which are polynomials of degree μ over polynomials of degree $d-1$ with coefficients in $\mathbb{Z}_q$. An element of $\mathbb{Z}_q$ is of bit-size $n\nu$, therefore the bit-size of the objects is $n\nu\mu d = O(n^{3+\varepsilon})$. Hence the $O(1)$ multiplications we have to do in the final iteration take time in $O(n^{3+\varepsilon})$. Applying **Red 1** to the result has the same asymptotic complexity (we have to visit the whole object and the runtime is linear in its size) but is faster in practice. Finally the overall complexity of Step 3 is in $O(n^{3+\varepsilon})$.

In Step 4 we do a constant number of multiplications (remember r is constant) and then an application of **Red 1** to objects of size in $O(n^{3+\varepsilon})$. Again the complexity is $O(n^{3+\varepsilon})$. Note that in the hyperelliptic case, this step does not exist.

In Step 5 we repeat a reduction process $2g$ times using **Red 2** and **Red 3**. More precisely, Substep 5.a is only reorganizing and applying **Red 1**; this takes negligible time. In Substep 5.b we repeat μ times a process which involves elementary operations over polynomials of degree at most d over $\mathbb{Z}_q$, i.e. a constant number of operations in $\mathbb{Z}_q$. Hence Substep 5.b has a cost in $O(n^{3+\varepsilon})$. The third reduction in Substep 5.c is negligible.

In Step 6 the costly part is to compute the norm of the matrix. By a recursive "divide and conquer" computation, we can save some of the costly Frobenius computations and obtain a runtime again in $O(n^{3+\varepsilon})$. In [31], Satoh proposes another method which can moreover save memory.

Putting everything together, the complexity of the algorithm is $O(n^{3+\varepsilon})$ in time and in space.

5 Numerical Results and Cryptographic Significance

As far as we know, even the original algorithm of Kedlaya has not yet been tested in practice. Therefore, we did our first implementation with the first aim of validating Kedlaya's algorithm and our extension. We used MAGMA, version 2.7, which allowed us to easily manipulate quite complicated objects: it is possible in MAGMA to construct the ring $\mathbb{Z}_q$ and to build the ring of series over polynomials over $\mathbb{Z}_q$ which is required. However, by taking such a high programming level, we can not really hope to do all the optimizations we could dream of; furthermore, there are some small bugs in our version of MAGMA which make us lose precision from time to time and we had to take a (constant) added margin in the precision of the computations. Therefore the results we give here are just meant to show that the algorithm works in practice and that cryptographic sizes are clearly reachable. We are currently working on an optimized implementation in C which should reduce the runtime significantly.

All the examples have been run on an Alpha EV6 at 667 MHz. The numbers of points are small enough that it is possible to factor them and prove the results. The space requirement was roughly 150 MB.

5.1 Hyperelliptic Examples

In the hyperelliptic case, we cannot take a field of characteristic 2 for which the algorithm is not designed. We carried out our experiments with finite fields of characteristic 3.

Example 1.

In $\mathbb{F}_{3^{53}}$, we take the generator t given by $t^{53} + 2t^4 + 2t^3 + 2t^2 + 1 = 0$ and consider the genus 2 randomly chosen curve given by

$$\begin{aligned} y^2 = x^5 &+ t^{2321121798755003703020989}x^4 + t^{8444066873716648223072527}x^3 + \\ &t^{7946343052437940195139141}x^2 + t^{10959512142684015392587300}x + \\ &t^{11366373156356845343093334}\,. \end{aligned}$$

After about 22 hours of computation we found the coefficients of its characteristic polynomial to be

$$\begin{aligned} a_1 &= 3767947898876, \\ a_2 &= 16462680188903823501200294, \end{aligned}$$

which yields a cardinality of

$$N = 375710212613709295385367112322529717794218564821248.$$

Example 2.

We took a randomly chosen curve over the finite field $\mathbb{F}_q$ with $q = 3^{37}$. Let t with minimal polynomial $t^{37} + t^3 + 2t^2 + 2t + 1$, and consider the genus 3 curve of equation

$$\begin{aligned} y^2 = x^7 &+ t^{145005605337803244}x^6 + t^{367106618571281107}x^5 + t^{377813655811225893}x^4 + \\ &t^{47288412099057887}x^3 + t^{55871015404698790}x^2 + t^{232037785016055219}x + \\ &t^{286815047052544398}\,. \end{aligned}$$

After about 30 hours of computation we found the coefficients of its characteristic polynomial to be

$$\begin{aligned}
a_1 &= 1128783670,\\
a_2 &= 1117168429648455309,\\
a_3 &= 886287268279616285414037148,
\end{aligned}$$

which yields a cardinality of

$$N = 912975818939808174202238856553992617331283588845689672.$$

5.2 Superelliptic Examples

For superelliptic curves, we concentrate on characteristic 2 which is the most interesting case for practical applications.

Example 1.
In $\mathbb{F}_{2^{53}}$, we take the generator t given by $t^{53}+t^6+t^2+t+1=0$ and consider the randomly chosen curve

$$\begin{aligned}
y^3 = {} & x^4 + t^{2256567407303775}x^3 + t^{7508555791178511}x^2 + \\
& t^{1136027055799467}x + t^{4967542575384673} .
\end{aligned}$$

After about 22 hours of computation we found the coefficients of its characteristic polynomial to be

$$\begin{aligned}
a_1 &= 0,\\
a_2 &= -2299871474212151,\\
a_3 &= 0,
\end{aligned}$$

which yields a cardinality of

$$N = 730750818665451438386441787834386121601727865546.$$

The nullity of a_1 and a_2 is not a surprise: it is explained by the absence of third roots of unity in the base field (see below).

Example 2.
In $\mathbb{F}_{2^{58}}$, we take the generator t given by $t^{58}+t^{19}+1=0$ and consider the random curve

$$\begin{aligned}
y^3 = {} & x^4 + t^{184416898722999862}x^3 + t^{138153554162118062}x^2 + \\
& t^{90053985362597546}x + t^{159188191651769175} .
\end{aligned}$$

After about 28 hours of computation we found the coefficients of its characteristic polynomial to be

$$\begin{aligned}
a_1 &= 1346491223,\\
a_2 &= 540650236559852363,\\
a_3 &= 106786896758507851646763008,
\end{aligned}$$

which yields a cardinality of

$$N = 2394524293789162792332288212231614478974438189795 4979.$$

The cardinalities we found were not (almost) prime and these curves should not be used in cryptography. We could have repeated the computations for several curves until we found a good curve. Note that an early-abort strategy cannot be used in this context.

5.3 Some Remarks about Superelliptic Curves in Cryptography

When one wants to build a cryptosystem based upon a curve, there are some security issues that have to be taken into account. Besides the fact that the number of points of the Jacobian should be (almost) prime, the following attacks (or threats) should be avoided:

1. Index-calculus attack for high genus curves [14, 4]: the genus of the curve should be at most 3.
2. MOV attack [25, 11]: the smallest k such that $\#\mathrm{J}(\mathcal{C}) \mid q^k - 1$ should be large.
3. Rück's attack [30]: the order of the subgroup in which we are working should be coprime to p.
4. The curve should not have "special properties".

Item 1 means that we are left with a small choice of non-elliptic curves useful for cryptography: hyperelliptic curves of genus 2 and 3, and superelliptic curves of the form $y^3 = f(x)$ with f of degree 4.

Items 2 and 3 are almost always fulfilled when we choose random curves and the verification that this is indeed the case for a given curve is straightforward.

The fourth item is less precise but has its importance: nowadays some people do not recommend to use elliptic curves for which the class number of the ring of endomorphism is too small; the base field should be a prime field or a prime extension field due to the threat of an attack by Weil descent [16]; and more generally any special behavior of the curves could be considered as suspect.

Keeping all this in mind, consider now a curve $\mathcal{C}$ of the form $y^3 = f(x)$ with f of degree 4 over a field $\mathbb{F}_q$. Assume that q is congruent to 2 modulo 3. Then every element of $\mathbb{F}_q$ is a cube. Therefore $\#\mathcal{C}/\mathbb{F}_q$ is equal to $q+1$, counting the point at infinity. Furthermore this is the case in every extension of $\mathbb{F}_q$ which does not contain the third roots of unity, namely every odd degree extension. A simple calculation with zeta functions shows that this implies that the coefficients a_1 and a_3 in the characteristic polynomial are zero, therefore $\chi(T)$ is of the form $T^6 + a_2T^4 + qa_2T^2 + q^3$, as we observed in Example 1. It means that this curve is highly "non-random" among all the curves of genus 3. In particular, the 3-torsion part of the Jacobian is partly degenerate which is a first step towards supersingularity. In [35], the reader will find a survey about the gradation from ordinary curves to supersingular curves and the link with the Newton polygon of the characteristic polynomial.

Having noticed this, one is tempted to claim that it is safer to take a base field which includes the third roots of unity. However, we are confronted to another problem, at least in characteristic 2. Indeed, $\mathbb{F}_{2^n}$ will contain the third root of unity if and only if n is even. We could then be subject to a Weil descent attack: if $n = 2m$, by doing a Weil restriction on $\mathrm{J}(\mathcal{C})$, we get an abelian variety of dimension 6 over $\mathbb{F}_{2^m}$. If someone is able to draw a curve of genus 6 on this abelian variety, then the system is broken. As far as we know, nobody is able to find such a curve (if it exists!) but this could be threatening enough to discourage the use $\mathcal{C}$ for cryptography.

This phenomenon is only true when one wants to use a base field of small characteristic. If we use a curve over a prime field, no Weil descent attack is to be feared and one can take a base field with roots of unity. This implies that there are additional automorphisms in the Jacobian and that the key-size should be slightly enlarged accordingly [8].

6 Conclusion

We have presented an extension of Kedlaya's algorithm in order to count points on superelliptic curves over finite fields of small characteristic. The time complexity is the same as the complexity for hyperelliptic curves. This complexity is asymptotically the same as the best known methods for counting points on elliptic curves. Note however that the ε which is involved in the expression $O(\log^{3+\varepsilon} q)$ does not hide the same logarithmic factors. We obtained some numerical examples proving that it is now feasible to count points of random hyperelliptic and superelliptic curves up to genus 3, for cryptographic sizes.

Further research topics are: extend Kedlaya's algorithm for hyperelliptic curves to characteristic 2, reduce the space complexity to $O(\log^2 q)$, extend the algorithm to C_{ab} curves or even to more general varieties (in fact Monsky–Washnitzer cohomology exists for more general varieties).

Acknowledgements

We are grateful to François Morain and to Guillaume Hanrot for their continuous support and many discussions during this work. A special thanks to Robert Harley for his close reading and his helpful comments.

Some of the computations were carried out on the machines of the UMS Medicis at École Polytechnique.

References

1. L. M. Adleman, J. DeMarrais, and M.-D. Huang. A subexponential algorithm for discrete logarithms over the rational subgroup of the jacobians of large genus hyperelliptic curves over finite fields. In L. Adleman and M.-D. Huang, editors,

ANTS-I, volume 877 of *Lecture Notes in Comput. Sci.*, pages 28–40. Springer-Verlag, 1994.
2. S. Arita. Algorithms for computations in Jacobians of C_{ab} curve and their application to discrete-log-based public key cryptosystems. In *Proceedings of Conference on The Mathematics of Public Key Cryptography, Toronto, June 12–17*, 1999.
3. S. Arita. Construction of secure C_{ab} curves using modular curves. In W. Bosma, editor, *ANTS-IV*, volume 1838 of *Lecture Notes in Comput. Sci.*, pages 113–126. Springer-Verlag, 2000.
4. S. Arita. Gaudry's variant against C_{ab} curve. In H. Imai and Y. Zheng, editors, *Public Key Cryptography*, volume 1751 of *Lecture Notes in Comput. Sci.*, pages 58–67. Springer-Verlag, 2000.
5. S. Arita. Weil descent of elliptic curves over finite fields of characteristic three. In T. Okamoto, editor, *Advances in Cryptolgy – ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Comput. Sci.*, pages 248–258. Springer-Verlag, 2000.
6. A. Basiri, A. Enge, J.-C. Faugère, and N. Gürel. Fast arithmetic on superelliptic cubics. In preparation.
7. I. Blake, G. Seroussi, and N. Smart. *Elliptic curves in cryptography*, volume 265 of *London Math. Soc. Lecture Note Ser.* Cambridge University Press, 1999.
8. I. Duursma, P. Gaudry, and F. Morain. Speeding up the discrete log computation on curves with automorphisms. In K.Y. Lam, E. Okamoto, and C. Xing, editors, *Advances in Cryptology – ASIACRYPT '99*, volume 1716 of *Lecture Notes in Comput. Sci.*, pages 103–121. Springer-Verlag, 1999.
9. M. Fouquet, P. Gaudry, and R. Harley. An extension of Satoh's algorithm and its implementation. *J. Ramanujan Math. Soc.*, 15:281–318, 2000.
10. M. Fouquet, P. Gaudry, and R. Harley. Finding secure curves with the Satoh-FGH algorithm and an early-abort strategy. In B. Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Comput. Sci.*, pages 14–29. Springer-Verlag, 2001.
11. G. Frey and H.-G. Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Math. Comp.*, 62(206):865–874, April 1994.
12. W. Fulton. *Algebraic curves.* Math. Lec. Note Series. W. A. Benjamin Inc, 1969.
13. S. Galbraith, S. Paulus, and N. Smart. Arithmetic on superelliptic curves. To appear Math. Comp.
14. P. Gaudry. An algorithm for solving the discrete log problem on hyperelliptic curves. In B. Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Comput. Sci.*, pages 19–34. Springer-Verlag, 2000.
15. P. Gaudry and R. Harley. Counting points on hyperelliptic curves over finite fields. In W. Bosma, editor, *ANTS-IV*, volume 1838 of *Lecture Notes in Comput. Sci.*, pages 313–332. Springer-Verlag, 2000.
16. P. Gaudry, F. Hess, and N. Smart. Constructive and destructive facets of Weil descent on elliptic curves. To appear in J. Crypt.
17. R. Harasawa and J. Suzuki. Fast jacobian group arithmetic on C_{ab} curves. In W. Bosma, editor, *ANTS-IV*, volume 1838 of *Lecture Notes in Comput. Sci.*, pages 359–376. Springer-Verlag, 2000.
18. K. Kedlaya. Counting points on hyperelliptic curves using Monsky-Washnitzer cohomology. Preprint, available at `http://arXiv.org/abs/math/0105031`, 2001.
19. N. Koblitz. *p-adic numbers, p-adic analysis and Zeta-functions*, volume 58 of *Graduate Texts in Mathematics*. Springer-Verlag, 1977.
20. N. Koblitz. *p-adic analysis: a short course on recent work*, volume 46 of *London Math. Lec. Note Series.* Cambridge University Press, 1980.

21. N. Koblitz. Elliptic curve cryptosystems. *Math. Comp.*, 48(177):203–209, January 1987.
22. N. Koblitz. Hyperelliptic cryptosystems. *J. of Cryptology*, 1:139–150, 1989.
23. R. Lercier. *Algorithmique des courbes elliptiques dans les corps finis.* Thèse, École polytechnique, June 1997.
24. D. Lorenzini. *An invitation to arithmetic geometry*, volume 106 of *Graduate Studies in Mathematics.* AMS, 1993.
25. A. Menezes, T. Okamoto, and S. A. Vanstone. Reducing elliptic curves logarithms to logarithms in a finite field. In *Proceedings 23rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 80–89. ACM Press, 1991. May 6–8, New Orleans, Louisiana.
26. V. Miller. Use of elliptic curves in cryptography. In A. M. Odlyzko, editor, *Advances in Cryptology – CRYPTO '86*, volume 263 of *Lecture Notes in Comput. Sci.*, pages 417–426. Springer-Verlag, 1987.
27. P. Monsky and G. Washnitzer. Formal cohomology: I. *Ann. of Math. (2)*, 88:181–217, 1968.
28. P. Monsky. Formal cohomology: III. *Ann. of Math. (2)*, 93:315–343, 1971.
29. J. Pila. Frobenius maps of abelian varieties and finding roots of unity in finite fields. *Math. Comp.*, 55(192):745–763, October 1990.
30. H. G. Rück. On the discrete logarithm in the divisor class group of curves. *Math. Comp.*, 68(226):805–806, 1999.
31. T. Satoh. Asymptotically fast algorithm for computing the Frobenius substitution and norms over unramified extension of p-adic number fields. Preprint 2001.
32. T. Satoh. The canonical lift of an ordinary elliptic curve over a finite field and its point counting. *J. Ramanujan Math. Soc.*, 15:247–270, 2000.
33. B. Skjernaa. Satoh's algorithm in characteristic 2. To appear in Math. Comp.
34. F. Vercauteren, B. Preneel, and J. Vandewalle. A memory efficient version of Satoh's algorithm. In B. Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Comput. Sci.*, pages 1–13. Springer-Verlag, 2001.
35. N. Yui. On the jacobian varietes of hyperelliptic curves over fields of characteristic $p > 2$. *J. Algebra*, 52:378–410, 1978.