Reynald LERCIER et Thomas SIRVENT

On Elkies subgroups of ℓ-torsion points in elliptic curves defined over a finite field

On Elkies subgroups of ℓ-torsion points in elliptic curves defined over a finite field

par REYNALD LERCIER et THOMAS SIRVENT

RÉSUMÉ. En sous-résultat de l'algorithme de Schoof-Elkies-Atkin pour compter le nombre de points d'une courbe elliptique définie sur un corps fini de caractéristique p, il existe un algorithme qui, pour ℓ un nombre premier d'Elkies, calcule des points de ℓ-torsion dans une extension de degré $\ell - 1$ à l'aide de $\tilde{O}(\ell \max(\ell, \log q)^2)$ opérations élémentaires à condition que $\ell \leqslant p/2$.

Nous combinons ici un algorithme rapide dû à Bostan, Morain, Salvy et Schost avec l'approche p-adique suivie par Joux et Lercier pour obtenir un algorithme valide sans limitation sur ℓ et p et de complexité similaire. Par soucis de simplification, nous décrivons précisément ici l'algorithme dans le cas des corps finis de caractéristique $p \geqslant 5$. Nous l'illustrons aussi avec quelques expérimentations.

ABSTRACT. As a subproduct of the Schoof-Elkies-Atkin algorithm to count points on elliptic curves defined over finite fields of characteristic p, there exists an algorithm that computes, for ℓ an Elkies prime, ℓ-torsion points in an extension of degree $\ell - 1$ at cost $\tilde{O}(\ell \max(\ell, \log q)^2)$ bit operations in the favorable case where $\ell \leqslant p/2$.

We combine in this work a fast algorithm for computing isogenies due to Bostan, Morain, Salvy and Schost with the p-adic approach followed by Joux and Lercier to get an algorithm valid without any limitation on ℓ and p but of similar complexity. For the sake of simplicity, we precisely state here the algorithm in the case of finite fields with characteristic $p \geqslant 5$. We give experiment results too.

1. Introduction

Let $\mathbf{K}$ be a finite field with q elements and E be an elliptic curve over $\mathbf{K}$ given by a plane equation of the form

$$(1.1) \qquad y^2 + a_1 xy + a_3 y = x^3 + a_2 x^2 + a_4 x + a_6$$

where the coefficients a_1, a_2, a_3, a_4 and a_6 are elements of $\mathbf{K}$. For any field $\mathbf{L}$ such that $\mathbf{K} \subset \mathbf{L}$, we denote by $E(\mathbf{L})$ the set of $\mathbf{L}$-points of E, $i.e.$ the set of solutions in $\mathbf{L}$ of Equation (1.1), plus the additional point at infinity O with homogeneous coordinates $(0 : 1 : 0)$. The curve $E/\mathbf{K}$ has a structure of commutative algebraic group with neutral element O, derived from the secant and tangent rules. Its order is equal to $q + 1 - t$ where the integer t satisfies $|t| \leqslant 2\sqrt{q}$.

We are interested in the determination of ℓ-torsion points of E, that is the set $E[\ell]$ of points P of $E(\overline{\mathbf{K}})$ such that $\ell P = O$ for prime integers ℓ, distinct from p. This group is isomorphic to $\mathbb{Z}/\ell\mathbb{Z} \times \mathbb{Z}/\ell\mathbb{Z}$ (cf.[18, p. 89]), its cardinality is thus ℓ^2. In fact, the multiplication by ℓ is given by a rational transformation of $\mathbb{P}^2(\mathbf{K})$, of degree ℓ^2, of the form $(x : y : z) \mapsto (X_\ell(x, y, z) : Y_\ell(x, y, z) : Z_\ell(x, y, z))$ where X_ℓ, Y_ℓ and Z_ℓ are three homogeneous polynomials of degree ℓ^2 and ℓ-torsion points are explicitly given by $Z_\ell(x, y, z) = 0$. Excluding the point O, this equation can be easily transformed into an equality of the form $f_\ell(x) = 0$ where f_ℓ is a monic univariate polynomial of degree $(\ell^2 - 1)/2$, called the ℓ-th division polynomial.

The improvements by Atkin and Elkies to Schoof's algorithm for counting points on elliptic curves stem from the fact that when the principal ideal (ℓ) splits in the imaginary quadratic field $\mathbb{Q}(\sqrt{t^2 - 4q})$, in half the cases thus, there exists two subgroups of degree ℓ in $E[\ell]$ defined in a degree $\ell - 1$ extension of $\mathbf{K}$. Such an integer ℓ is classically called an Elkies prime, and similarly, we call these two subgroups ℓ-th Elkies subgroups.

In this work, we focus on algorithmic efficient ways to compute degree $(\ell - 1)/2$ polynomials, defined over $\mathbf{K}$, the roots of which are abscissas of points contained in an ℓ-th Elkies subgroup. We call these polynomials ℓ-th Elkies polynomials too. Our main result, where we classically denote by $\phi_1(x_1, \ldots, x_k) = \tilde{O}(\phi_2(x_1, \ldots, x_k))$ functions ϕ_1 and ϕ_2 such that there exists an integer k with $\phi_1(x_1, \ldots, x_k) = O(\phi_2(x_1, \ldots, x_k)\log^k \phi_2(x_1, \ldots, x_k))$, is as follows.

Theorem 1. *Let E be a generic elliptic curve defined over a finite field $\mathbf{K}$ with q elements and characteristic $p \geqslant 5$ and ℓ be an Elkies prime, distinct from the characteristic of $\mathbf{K}$, then Algorithm 2 computes ℓ-th Elkies polynomials at cost $\tilde{O}(\ell \max(\ell, \log q)^2)$ bit operations and space.*

The "generic" condition for the elliptic curve E in Theorem 1 means that, if we pick at random a curve, the algorithm almost always returns the correct result. Especially, it is very unlikely that the algorithm encounters issues on ordinary elliptic curves with a large discriminant (see the beginning of Section 2 for a discussion on this topic).

Furthermore, Algorithm 2 (*cf.* Section 4) takes in input curves defined over finite fields of characteristic $p \geqslant 5$ by a Weierstrass equation of the form $y^2 = x^3 + a_4 x + a_6$. For $p = 2$ (or $p = 3$), Weierstrass models of the form $y^2 + xy = x^3 + a_2 x^2 + a_6$ (or $y^2 = x^3 + a_2 x^2 + a_6$) must be considered. This yields completely different equations (see for instance [11, 12] in the case $p = 2$). Theorem 1 can be easily extended to these fields but for the sake of simplicity we prefer to omit the details here.

This problem is closely related to the problem of computing separable isogenies of degree ℓ between two elliptic curves since an application of Velu's formulas [20] with starting point such polynomials yields an isogeny. Especially, counting points on elliptic curves first raised interest for such computations. But isogenies now play a role in numerous other fields, for instance to protect elliptic curve cryptographic devices against physical side attacks [19], to improve Weil descent to calculate elliptic discrete logarithms [10], to decrease the complexity of computing discrete logarithms in some family of finite fields [7], to exhibit normal basis in finite field extensions [6], etc.

We first recall in Section 2 the complexity of the algorithms known to solve this problem. In Section 3, we focus on the fastest algorithm in finite fields of large characteristic published so far, due to Bostan, Morain, Salvy and Schost [2]. We then show in Section 4 how we can combine this algorithm with the p-adic approach introduced by Joux and Lercier in [11] to get a fast algorithm in any finite field and we clarify that we need a p-adic precision of only $O(\log^2 \ell / \log p)$. A detailed example is given in Section 5.

2. Related work

We restrict ourself to finite fields $\mathbf{K}$ of characteristic $p \geqslant 5$ and to prime integers $\ell > 2$. In this case, an elliptic curve E is simply given by a plane equation of the form $y^2 = x^3 + a_4 x + a_6$. Its discriminant, always non zero, is equal to $\Delta_E = -16(4a_4{}^3 + 27a_6{}^2)$ and its j-invariant is equal to $j_E = -12^3 (4a_4)^3 / \Delta_E$.

We moreover rely upon Schoof-Elkies-Atkin's algorithm, and we thus assume in the remaining of the paper that E is ordinary with j-invariant $j_E \neq 0, 1728$ [16]. If we denote $j_{E'}$, the j-invariant of an elliptic curve E' that is ℓ-isogenous to E, we furthermore needs that $(j_E, j_{E'})$ is not a singular point of the modular curve $X_0(\ell)$. We refer to [16, pages 248–249] for a detailed discussion on this phenomenon. Especially, except elliptic curves with very small discriminants, it is very unlikely that ordinary elliptic curves behave badly.

2.1. Naive approach. ℓ-th Elkies polynomials are factors of the ℓ-th division polynomial f_ℓ. Therefore, a naive approach consists in computing f_ℓ,

which can be done at cost $\tilde{O}(\ell^2 \log q)$ elementary operations thanks to a "Square and Multiply" method [18], and then in factorizing it with cost $\tilde{O}(\ell^{1.815\times 2} \log^2 q)$ [17]. This algorithm needs a total of $\tilde{O}(\ell^{3.63} \log^2 q)$ bit operations.

2.2. Schoof-Elkies-Atkin framework. Let π_E be the Frobenius endomorphism of E. Its restriction to $E[\ell]$, seen as a $\mathbb{F}_\ell$-vector space of dimension two, is still an endomorphism. When ℓ is an Elkies prime, its eigenspaces correspond to ℓ-th Elkies subgroups C of $E[\ell]$ and from each C one can construct a normalized isogeny of degree ℓ between E and an elliptic curve **K**-isomorphic to $E' = E/C$.

The following method takes advantage of these facts.

Step 1: Compute the modular polynomial of degree ℓ, $\Phi_\ell(X, Y)$, equation of the modular curve $X_0(\ell)$. This is a bivariate symmetric polynomial, of degree $\ell + 1$ in X and Y, whose coefficients are integers of $\tilde{O}(\ell)$ bits (*cf.* [4]). j-invariants of ℓ-isogenous elliptic curves are roots of $\Phi_\ell(X, Y)$.

Step 2: Compute roots $j_{E'}$ and $j_{E''}$ of $\Phi_\ell(X, j_E)$.

Step 3: Compute a normalized Weierstrass equation for elliptic curves E' and E'' of j-invariants $j_{E'}$ and $j_{E''}$, and the sums of the abscissas of points in the kernel of the isogenies $E \to E'$ and $E \to E''$, using the polynomials Φ_ℓ, $\partial\Phi_\ell/\partial X$, $\partial\Phi_\ell/\partial Y$, $\partial^2\Phi_\ell/\partial X^2$, $\partial^2\Phi_\ell/\partial X\partial Y$, $\partial^2\Phi_\ell/\partial Y^2$ (*cf.* [16]).

Step 4: Compute from each isogenous curve, a ℓ-th Elkies polynomial thanks to the kernel of the corresponding isogeny.

The complexity analysis comes now.

Step 1: The modular polynomial $\Phi_\ell(X, Y)$ has $O(\ell^2)$ coefficients, each with about $\tilde{O}(\ell)$ bits. There exists methods to compute this polynomial at cost quasi-linear in its size, *i.e.* in $\tilde{O}(\ell^3)$ bit operations (*cf.* [9]). We need to reduce this polynomial modulo p, that is $\tilde{O}(\ell^3)$ bit operations too. The result is then of size $O(\ell^2 \log p)$ bits.

Step 2: With the help of Horner's method, the evaluation of $\Phi_\ell(X, Y)$ at j_E costs $\tilde{O}(\ell^2 \log q)$ bit operations. In order to compute roots of the resulting degree $\ell + 1$ polynomial, we have first to compute its gcd with $X^q - X$, that is $\tilde{O}(\ell \log^2 q)$ bit operations (*cf.* [13]). We obtain a degree 2 polynomial whose roots can then be found with negligible cost.

Step 3: The computations of the derivatives of Φ_ℓ and their evaluations can be done at cost $\tilde{O}(\ell^2 \log q)$ bit operations.

Step 4: Here, we have to distinguish several cases.

- In finite fields of large characteristic, the best algorithm known so far to compute isogenies is due to Bostan et al. [2] and takes time $\tilde{O}(\ell \log q)$ bit operations.
- In finite fields of small but *fixed* characteristic, the best algorithm known is due to Couveignes [5] and needs $\tilde{O}(\ell^2 \log q)$ bit operations (but the contribution of p in the $\tilde{O}$ complexity constant is exponential in $\log p$).
- In between, that is finite fields of small but *non-fixed* characteristic, the best algorithm is due to Joux and Lercier [11] and needs $\tilde{O}((1 + \ell/p)\, \ell^2 \log q)$ bit operations.

The best total complexity is thus equal to $\tilde{O}(\ell \max(\ell, \log q)^2)$, achieved in finite fields of large characteristic. But, in finite fields of small characteristic, the complexity can be as large as $\tilde{O}(\ell^3 \log q)$ bit operations when $\ell \gg p$.

This work yields an algorithm of same complexity as in the large characteristic case without any limitation on the characteristic or the degree of the base field $\mathbf{K}$.

3. The large characteristic case

In order to get an algorithm with good complexity in finite fields of small characteristic too, we first reformulate the algorithm of Bostan et al. in such a way that its extension in the p-adics is more easily studiable. The general strategy is the same except that we take into account some specificities of the involved differential equation in the resolution. As a result, we obtain a precise and compact algorithm (*cf.* Algorithm 1).

3.1. Differential equation. In a field $\mathbf{K}$ of characteristic larger than three, an isogeny between two elliptic curves, $E : y^2 = x^3 + a_4\, x + a_6$ and $E' : y^2 = x^3 + a_4'\, x + a_6'$, can be given by

$$I(x, y) = \left(\frac{N(x)}{D(x)}, cy \left(\frac{N(x)}{D(x)} \right)' \right),$$

where N and D are monic polynomials of degree ℓ and $\ell - 1$. When c is equal to one, the isogeny is said to be normalized. This is in particular the case in the Schoof–Elkies–Atkin framework.

If we now state that the image of a point of E by I is on E', we get the following differential equation

$$(3.1) \qquad (x^3 + a_4\, x + a_6) \left(\frac{N(x)}{D(x)} \right)'^2 = \left(\frac{N(x)}{D(x)} \right)^3 + a_4' \left(\frac{N(x)}{D(x)} \right) + a_6'.$$

This equation can be solved with a Taylor series expansion of $N(x)/D(x) - x$ in $1/x$ for $1/x$ close to 0. The relations obtained thanks to Equation (3.1)

enable to compute by recurrence each coefficient in turn, if the first coefficients are known. It is then possible to recover N and D with the help of Berlekamp-Massey's algorithm, or one of its optimized variant. In [2], one takes advantage of a Newton algorithm so that the number of coefficients computed at each iteration doubles.

More precisely, let S be defined by

$$S(x) = \sqrt{\frac{D\left(1/x^2\right)}{N\left(1/x^2\right)}}, \quad \text{or equivalently} \quad \frac{N(x)}{D(x)} = \frac{1}{S\left(1/\sqrt{x}\right)^2}.$$

The square root is chosen such that $S(x) = x + O(x^3)$ at infinity, which is possible because $N(x)/D(x)$ has a series expansion of the form $x + O(1)$. Equation (3.1) becomes then

$$(a_6 x^6 + a_4 x^4 + 1)S'(x)^2 = 1 + a_4' S(x)^4 + a_6' S(x)^6.$$

This is enough to determine $S(x)$, and finally recover $N(x)/D(x)$.

3.2. Resolution. We consider more generally equations of the form $S'^2 = G \cdot (H \circ S)$. In Equation (3.1), we have for instance $H(z) = a_6' z^6 + a_4' z^4 + 1$ and $G(x) = 1/(a_6 x^6 + a_4 x^4 + 1)$. We now look for a solution modulo x^μ, where μ is an integer given in input. The way to solve this equation is first to assume that we know the solution modulo x^d and then, thanks to a Newton iteration, to obtain a solution modulo x^{2d}. After roughly $\log \mu$ such iterations, one gets the full solution.

We now present Algorithm 1, a compact algorithm for this task (where $\int$ denotes an integral operation with integration constant equal to zero). Its complexity can be easily established, it is equal to $\tilde{O}(\mu \log q)$ bit operations. Its correctness is slightly more difficult to prove and we delay it to Appendix A.

Algorithm 1 Solving equation $S'^2 = G \cdot (H \circ S)$, $S(0) = \alpha$ and $S'(0) = \beta$.

Input: $\mu \in \{1, \ldots, p\}$, $(\alpha, \beta) \in \mathbf{K} \times \mathbf{K}^*$, $H \in \mathbf{K}[z]$, $G \in \mathbf{K}[[x]]$
Output: $S \in \mathbf{K}[x]$, a solution of the differential equation modulo x^μ

  $d \longleftarrow 2, \; U \longleftarrow 1/\beta, \; J \longleftarrow 1, \; V \longleftarrow 1$
  $S \longleftarrow \alpha + \beta \, x + \left[(G'(0) + H'(\alpha) \, \beta^3)/(4\beta)\right] x^2$
  while $(d < \mu - 1)$ **do**
    $U \longleftarrow U \cdot (2 - S' \cdot U) \bmod x^d$
    $V \longleftarrow \left(V + J \cdot (H \circ S) \cdot (2 - V \cdot J)\right) / 2 \bmod x^d$
    $J \longleftarrow J \cdot (2 - V \cdot J) \bmod x^d$
    $S \longleftarrow S + V \cdot \int \left(G \cdot (H \circ S) - S'^2\right) (U \cdot J / 2) \, dx \bmod x^{\min(2d+1,\mu)}$
    $d \longleftarrow 2d$
  end while
  return S

Proposition 3.1. *Let $(\alpha, \beta) \in \mathbf{K} \times \mathbf{K}^*$ where $\mathbf{K}$ is a finite field of characteristic p, let G be a formal series defined over $\mathbf{K}$, let H be a polynomial defined over $\mathbf{K}$ such that $H(\alpha) = 1$ and $G(0) = \beta^2$. Let $\mu \in \{1, \ldots, p\}$, then Algorithm 1 computes a Taylor series (modulo x^μ) of the solution S of the differential equation*

$$(3.2) \qquad S'(x)^2 = G(x)\, H(S(x))\,, \; S(0) = \alpha\,, \; S'(0) = \beta.$$

3.3. Full algorithm. We first compute $G(x) = 1/(a_6\, x^6 + a_4\, x^4 + 1)$ modulo $x^{4\ell-1}$ thanks to the classical iterative following formula, $G_1(x) = 1$, $G_{2d}(x) = G_d(x)\,(2 - G_d(x) \cdot (a_6\, x^6 + a_4\, x^4 + 1)) \bmod x^{2d}$. We then apply Algorithm 1 to $G(x)$ and $H(z) = a_6'\, z^6 + a_4'\, z^4 + 1$ with $\mu = 4\ell$, $\alpha = 0$ and $\beta = 1$.

The obtained solution S is odd, we define from it

$$T(x) = \sum_{i=0}^{2\ell-1} t_i\, x^i\,, \text{ where } \forall i \in \{0, \ldots, 2\ell - 1\},\; t_i = s_{2i+1}.$$

We denote by $R(x)$ the inverse of the square of $T(x)$, modulo $x^{2\ell}$, with the same inverse formulas as those used for G. We then have

$$\frac{N(x)}{D(x)} = x\, R\left(\frac{1}{x}\right)\,,\; \text{ i.e. } R(x) = \frac{x^\ell\, N(1/x)}{x^{\ell-1}\, D(1/x)}\,.$$

Applying Berlekamp-Massey algorithm [1, 14, 8] or one of its optimized variant [3, 15] to R yields D and the searched ℓ-th Elkies polynomial is equal to the monic square root of D.

4. Extension to any finite field

To extend the Schoof-Elkies-Atkin framework in any characteristic, the techniques developed in [11] give the general idea: to use the p-adics to authorize divisions by the characteristic p of the field. These divisions make it possible to use in any finite field algorithms primarily designed in large characteristic. There exists one main obstacle with this approach. Calculations in the p-adics imply losses of precision at the time of divisions by p. It is thus necessary to anticipate a sufficient precision, which results in an increase in the size of the handled objects.

One could hope to perform this lift in the p-adics only at Step 4 of the Schoof-Elkies-Atkin method, *i.e.* for the calculation of the isogeny with a p-adic extension of Algorithm 1. It is actually not possible because fast algorithms for computing isogenies need normalized models for the isogenous curves.

It is thus necessary to lift in the p-adics from the very beginning of the algorithm. It is exactly what is done in [11], with a p-adic precision linear

in ℓ. Instead, we consider here the techniques of [2], and one shows that the necessary p-adic precision can be brought back to only $O(\log^2 \ell / \log p)$.

   This yields Algorithm 2. The total complexity of this algorithm is similar to the one of the large characteristic case, that is $\tilde{O}(\ell \max(\ell, \log q)^2)$.

Algorithm 2 Computing ℓ-th Elkies polynomials (p-adic Schoof-Elkies-Atkin framework)

Input: $E/\mathbb{F}_q : y^2 = x^3 + a_4 x + a_6$ a generic elliptic curve and ℓ an odd prime integer.

Output: An ℓ-th Elkies polynomial if there exists one.

   Step 1. Compute over $\mathbb{Z}$ the bivariate modular polynomial of degree ℓ, $\Phi_\ell(X, Y)$.

   Step 2. Compute at precision $\kappa = 1 + \lfloor (\log_2(4\ell - 1) + 1)^2 / \log_2 p \rfloor$, if it exists, roots $\tilde{j}_{E'}$ and $\tilde{j}_{E''}$ of $\Phi_\ell(X, \tilde{j}_E)$ where $\tilde{j}_E$ is the j-invariant of the elliptic curve $\tilde{E} : y^2 = x^3 + \tilde{a}_4 x + \tilde{a}_6$ with $\tilde{a}_4$ and $\tilde{a}_6$ any lifts in the p-adics of a_4 and a_6. Otherwise, return FAIL and STOP.

   Step 3. Compute normalized Weierstrass equations $y^2 = x^3 + \tilde{a}'_4 x + \tilde{a}'_6$ and $y^2 = x^3 + \tilde{a}''_4 x + \tilde{a}''_6$ for elliptic curves of j-invariant $\tilde{j}_{E'}$ and $\tilde{j}_{E''}$, and the sum of the abscissas of points in the kernel of the isogenies $E \to E'$ and $E \to E''$, using the polynomials Φ_ℓ, $\partial\Phi_\ell/\partial X$, $\partial\Phi_\ell/\partial Y$, $\partial^2\Phi_\ell/\partial X^2$, $\partial^2\Phi_\ell/\partial X\partial Y$, $\partial^2\Phi_\ell/\partial Y^2$.

   Step 4. For the curve E' (resp. E''):
   (1) Compute $G(x) = 1/(\tilde{a}_6 x^6 + \tilde{a}_4 x^4 + 1) \bmod x^{4\ell-1}, \ p^\kappa$.
   (2) Apply Algorithm 1 to $G(x)$ and $H(z) = \tilde{a}'_6 z^6 + \tilde{a}'_4 z^4 + 1$ with $\mu = 4\ell$, $\alpha = 0$ and $\beta = 1$ to obtain a Taylor expansion $\sum \tilde{s}_i x^i$ of the solution of the differential equation (3.2).
   (3) Compute $T(x) = \sum_{i=0}^{2\ell-1} t_i x^i$, where $\forall i \in \{0, \dots, 2\ell - 1\}$, $t_i = \tilde{s}_{2i+1} \bmod p$.
   (4) Compute $R(x) = 1/T(x)^2 \bmod x^{2\ell}, \ p$.
   (5) Apply a fast version of Berlekamp-Massey algorithm to find N and R, two polynomials such that $N(x)/D(x) = x\,R(1/x) \bmod p$.
   (6) Return the monic square root of $D(x) \bmod p$.

4.1. Lifting curves and isogenies. One starts by lifting arbitrarily the curve E in the p-adics. Any coefficient $\tilde{a}_4$ and $\tilde{a}_6$ such that $\tilde{a}_4 = a_4 \bmod p$ and $\tilde{a}_6 = a_6 \bmod p$ is appropriate and one works on an elliptic curve $\tilde{E}$ with model $y^2 = x^3 + \tilde{a}_4 x + \tilde{a}_6$.

   The computation of the j-invariant $\tilde{j}_E$ of the curve $\tilde{E}$, of the solutions $\tilde{j}_{E'}$ and $\tilde{j}_{E''}$ of the equation $\Phi_\ell(x, \tilde{j}_E) = 0$, as well as Weierstrass models of the corresponding curves $\tilde{E}'$ and $\tilde{E}''$, proceeds exactly as in the SEA

framework. The curves $\tilde{E}'$ and $\tilde{E}''$ are ℓ-isogenous with the curve $\tilde{E}$, and the isogenies can be calculated as in the large characteristic case.

Projection E' of the curve $\tilde{E}'$ on the base field $\mathbf{K}$ is ℓ-isogenous with E, and the connecting isogeny is the projection on the base field of the isogeny connecting $\tilde{E}$ to $\tilde{E}'$. It is the same for E''. It is thus enough to project the denominators of the isogenies on $\mathbf{K}$ to identify the required factors of the ℓ-th division polynomial of E.

4.2. p-adic computations. From now on, we are interested in the p-adic precision of the lift of the elliptic curve E. This precision must be large enough so that at the end of the resolution of the differential equation with Algorithm 1, the result S can be reduced in $\mathbf{K}$.

To this purpose, we need first some definitions.

Definition. *For any positive integer r, one defines $v_p(r)$ by the largest power of p which divides r, $v_p(r) = \max\left\{k \in \mathbb{N} \,|\, p^k \ \text{divides} \ r\right\}$.*

We denote by $\mathrm{Loss}(p, \ell)$ the sum $\sum_{1 \leqslant i < \log_2(4\ell-1)} \mathrm{Decrease}(p, \ell, i)$, where

$$\mathrm{Decrease}(p, \ell, i) \;=\; \max\left\{v_p(r) \,|\, 2^i + 1 \;\leqslant\; r \;\leqslant\; \min(2^{i+1}, 4\ell - 1)\right\}.$$

The following lemma relates the precision needed to the function Loss.

Lemma 4.1. *Let μ be the p-adic precision of the coefficients $\tilde{a}_4$ and $\tilde{a}_6$, then when $\mu > \mathrm{Loss}(p, \ell)$ the polynomials U, V, J and S computed in Algorithm 1 have p-adic integer coefficients. Furthermore the precision of the result S is at least equal to $(\mu - \mathrm{Loss}(p, \ell))$.*

Proof. One proves this theorem by recurrence on j, the number of iterations of the loop "while" in Algorithm 1. We assume that at rank j, $0 \leqslant j < \log_2(4\ell-1)$, the polynomials U, V, J and S have p-adic integer coefficients and that their precision is at least equal to $\mu - \sum_{1 \leqslant i \leqslant j} \mathrm{Decrease}(p, \ell, i)$.

INITIALIZATION. In input of the algorithm, we have $\alpha = 0$, $\beta = 1$, $H(z) = \tilde{a}_6' z^6 + \tilde{a}_4' z^4 + 1$ and $G(x) = 1/(\tilde{a}_6 x^6 + \tilde{a}_4 x^4 + 1)$. The elements $\tilde{a}_4$, $\tilde{a}_6$, $\tilde{a}_4'$ and $\tilde{a}_6'$ are integers of precision μ and thus G and H are of precision μ too (no division by p occurs in the computation of G). The same is true for U, V, J and S.

HEREDITY. Let $j < \log_2(4\ell - 1)$, we suppose the assumption true at rank $j - 1$. At the j-th iteration, polynomials U, V and J are updated via additions, multiplications, derivations and compositions of the values of U, V, J and S before the entry in the loop. All these operations preserve the precision and the polynomials U, V and J have p-adic integer coefficients with precision at least equal to $\mu - \sum_{1 \leqslant i \leqslant j-1} \mathrm{Decrease}(p, \ell, i)$.

For S, except the integral operation, the calculations preserve the precision. Coefficients of the series after the integral operation are inverses of degrees between $2^j + 1$ and $\min(2^{j+1}, 4\ell - 1)$. The largest power of p by which

we carry out a division is thus Decrease(p, ℓ, j). The absolute precision of the coefficients of S is thus higher or equal to $\mu - \sum_{1 \leqslant i \leqslant j}$ Decrease(p, ℓ, i). Furthermore, since this precision is positive, each coefficient of S is a lift of the coefficient of the series deduced from the isogeny over $\mathbf{K}$, and these coefficients are p-adic integers. $\qquad \square$

Lemma 4.2 yields a clear asymptotic bound on the loss of precision stated in Lemma 4.1.

Lemma 4.2. *We have* Loss$(p, \ell) = O\left(\log^2 \ell / \log p\right)$.

Proof. For all $i < \log_2(4\ell - 1)$, Decrease(p, ℓ, i) is the largest power of p which divides a range of integers, at most equal to 2^{i+1}, we have therefore Decrease$(p, \ell, i) \leqslant \log_p 2^{i+1}$, and

$$
\begin{aligned}
\text{Loss}(p, \ell) &\leqslant \log_p 2 \left(\sum_{1 \leqslant i < \log_2(4\ell - 1)} (i + 1) \right), \\
&\leqslant \log_p 2 \, \log_2(4\ell - 1) \, (\log_2(4\ell - 1) + 1) , \\
&\leqslant \left(\log_2(4\ell - 1) + 1 \right)^2 / \log_2 p .
\end{aligned}
$$

$\qquad \square$

Computations performed in the Schoof-Elkies-Atkin framework, especially calls to Algorithm 1, are thus realized in the p-adics with precision at most $O(\log^2 \ell / \log p)$. This precision does not modify the $\tilde{O}$ complexities of the large characteristic case and we still have in the p-adic case a total complexity equal to $\tilde{O}(\ell \max(\ell, \log q)^2)$ bit operations, as announced in Theorem 1.

5. Experiments

We have implemented this algorithm in the computer algebra system MAGMA. Thanks to it, we were able to observe that the bound on the precision stated in Lemma 4.1 is tight. We illustrate the method with an example too.

5.1. p-adic precision. Figure 1 shows the evolution of the precision when p and ℓ vary. The "The(oretical)" bound mentioned corresponds to $1 +$ Loss(p, ℓ) calculations. The "Obs(erved)" bound is what seems necessary at the time of calculations (checked on some examples).

It turns out that the precision observed in practice is near the theoretical bound. For many values of ℓ, a gap between the theoretical bound and the observed bound appears, but this difference remains quite small.

$p = 5$		
ℓ	Obs.	The.
7	5	5
11	6	6
13	6	7
17	7	8
19-31	8	9
37	11	11
41-61	11	12
67	13	14
71-89	13	15
97	14	16
131	16	17
257	21	22

$p = 7$		
ℓ	Obs.	The.
11	4	5
13	5	6
17	6	6
19-23	6	7
29-31	6	8
37-61	8	10
67-73	10	11
79-83	10	12
89-97	11	13
131	13	14
257	15	16

$p = 11$		
ℓ	Obs.	The.
13	3	4
17-29	4	5
31	5	6
37-59	6	7
61	6	8
67-89	7	9
97	8	10
131	9	11
257	12	12

FIGURE 1. p-adic precisions for $p = 5, 7, 11$ and $\ell \leqslant 257$.

5.2. Example. Let $E : y^2 = x^3 + x + 4$ be defined over $\mathbb{F}_5$ and $\ell = 11$. We first need to compute an upper bound for the 5-adic precision,

$$\text{Decrease}(5, 11, 1) = 0, \quad \text{Decrease}(5, 11, 2) = 1, \quad \text{Decrease}(5, 11, 3) = 1,$$
$$\text{Decrease}(5, 11, 4) = 2, \quad \text{Decrease}(5, 11, 5) = 1.$$

We find $\text{Loss}(5, 11) = 5$ and the 5-adic precision is thus 6.

A 5-adic lift of the curve at precision 6 is $y^2 = x^3 + x + 4 + O(5^6)$. With the help of the classical 5-th modular polynomial Φ_{11}, we find that a 11-isogenous curve is given by $y^2 = x^3 - 7329x - 3934 + O(5^6)$.

We can now compute the series $G(x)$ modulo $x^{4\ell-1}$.

$$G(x) = 4374x^{42} + 4298x^{40} - 2331x^{38} - 4417x^{36} + 3936x^{34} + 3505x^{32}$$
$$+ 228x^{30} - 1041x^{28} - 616x^{26} + 97x^{24} + 236x^{22} + 95x^{20} - 48x^{18}$$
$$- 47x^{16} - 12x^{14} + 15x^{12} + 8x^{10} + x^8 - 4x^6 - x^4 + 1 + O(5^6) \bmod x^{43}.$$

A solution of the differential equation based on $G(x)$ and $H(z) = \tilde{a}_6' z^6 + \tilde{a}_4' z^4 + 1$ is then given modulo x^{44} by

$$S(x) = -(2 + O(5)) x^{43} + (2 + O(5)) x^{41} - (1 + O(5)) x^{39} + (8 + O(5^2)) x^{37}$$
$$- (1 + O(5)) x^{35} + (O(5^2)) x^{33} + (O(5^2)) x^{31} - (10 + O(5^2)) x^{29} - (7 + O(5^2)) x^{27}$$
$$- (1 + O(5^2)) x^{25} + (192 + O(5^4)) x^{23} + (125 + O(5^4)) x^{21} + (293 + O(5^4)) x^{19}$$
$$+ (4 + O(5^4)) x^{17} - (161 + O(5^4)) x^{15} - (611 + O(5^5)) x^{13} + (211 + O(5^5)) x^{11}$$
$$- (1494 + O(5^5)) x^9 + (1058 + O(5^5)) x^7 - (733 + O(5^5)) x^5 + (O(5^6)) x^3 + (1 + O(5^6)) x,$$

and modulo 5, we find

$$T(x) = 3\,x^{21} + 2\,x^{20} + 4\,x^{19} + 3\,x^{18} + 4\,x^{17} + 3\,x^{15} + 3\,x^{13} + 4\,x^{12} + 2\,x^{11}$$
$$+\, 3\,x^9 + 4\,x^8 + 4\,x^7 + 4\,x^6 + x^5 + x^4 + 3\,x^3 + 2\,x^2 + 1 \bmod x^{22}\,.$$

We have $R(x) = 1/T(x)^2 \bmod x^{2\ell}$, that is

$$R(x) = 2\,x^{20} + 2\,x^{19} + 3\,x^{18} + x^{16} + 2\,x^{15} + 3\,x^{14} + x^{13} + 3\,x^{12} + 2\,x^{11}$$
$$+\, 2\,x^{10} + 2\,x^8 + 3\,x^7 + 4\,x^6 + 4\,x^5 + 4\,x^3 + x^2 + 1 \bmod x^{22}\,.$$

The rebuilding of the rational fraction corresponding to R gives

$$R(x) = \frac{3\,x^{11} + x^9 + x^8 + x^7 + x^6 + 3\,x^5 + 2\,x^4 + 3\,x^3 + 2\,x^2 + 2\,x + 1}{x^{10} + x^9 + x^8 + x^7 + 3\,x^6 + 3\,x^5 + 3\,x^4 + 2\,x^3 + x^2 + 2\,x + 1} \bmod x^{22}.$$

One reverses the order of the coefficients of the denominator to obtain

$$D(x) = x^{10} + 2\,x^9 + x^8 + 2\,x^7 + 3\,x^6 + 3\,x^5 + 3\,x^4 + x^3 + x^2 + x + 1.$$

The ℓ-th Elkies polynomial is then

$$\sqrt{D(x)} = x^5 + x^4 + x^2 + 3\,x + 1.$$

Appendix A. Proof of Proposition 3.1

Let d be a non-zero even integer, we assume that we know a solution of the differential equation modulo x^{d+1}. We thus have

(A.1) $$S_d'^{\,2} = G \cdot (H \circ S_d) \bmod x^d\,,\ S_d(0) = \alpha\,,\ S_d'(0) = \beta.$$

Let $S_{2d} = S_d + A_{2d}$ be a solution modulo x^{2d+1}, with x^{d+1} dividing A_{2d}, therefore $\left(S_d' + A_{2d}'\right)^2 = G \cdot \left(H \circ (S_d + A_{2d})\right) \bmod x^{2d}$. This yields a linear differential equation in A_{2d}.

$$2\,S_d' \cdot A_{2d}' - G \cdot (H' \circ S_d) \cdot A_{2d} = G \cdot (H \circ S_d) - S_d'^{\,2} \bmod x^{2d}\,.$$

With initial condition $A_{2d}(0) = 0$, a solution of this equation is

(A.2) $$A_{2d} = \frac{1}{J_{2d}} \int \frac{\left(G \cdot (H \circ S_d) - S_d'^{\,2}\right) \cdot J_{2d}}{2\,S_d'}\ dx \bmod x^{2d+1}\,,$$

$$\text{where}\quad J_{2d} = \exp\left(-\int \frac{G \cdot (H' \circ S_d)}{2\,S_d'}\ dx\right) \bmod x^{2d+1}\,.$$

From Eq. (A.1), we know that $\left(G \cdot (H \circ S_d) - S_d'^{\,2}\right)$ is divisible by x^d. Moreover, S_d' has a non-zero constant coefficient. A factor x^d appears then in the integral of A_{2d} and it's enough to compute J_{2d} modulo x^d. The inverse of J_{2d} is multiplied by the integral, it will thus be multiplied by x^{d+1}, and it's enough to evaluate this inverse modulo x^d. The inverse of S_d' is needed in the computations of A_{2d} and J_{2d}. In A_{2d}, this inverse is multiplied by x^d and we then compute a primitive. In J_{2d}, we compute

only modulo x^d. In both cases, the inverse of S'_d modulo x^d is enough. This inverse is provided by Eq. (A.1):

$$\frac{1}{S'_d} = \frac{S'_d}{G \cdot (H \circ S_d)} \bmod x^d.$$

We plug this expression in the computation of J_{2d} modulo x^d, we find

$$\int \frac{G \cdot (H' \circ S_d)}{2\, S'_d}\, dx = \int \frac{S'_d \cdot (H' \circ S_d)}{2\,(H \circ S_d)}\, dx \bmod x^d$$

$$= \frac{\log(H \circ S_d)}{2} \bmod x^d.$$

We then find the following nice formulas for J_{2d} and $1/J_{2d}$ modulo x^d,

$$J_{2d} = \frac{1}{\sqrt{H \circ S_d}} \bmod x^d, \quad \frac{1}{J_{2d}} = \sqrt{H \circ S_d} \bmod x^d.$$
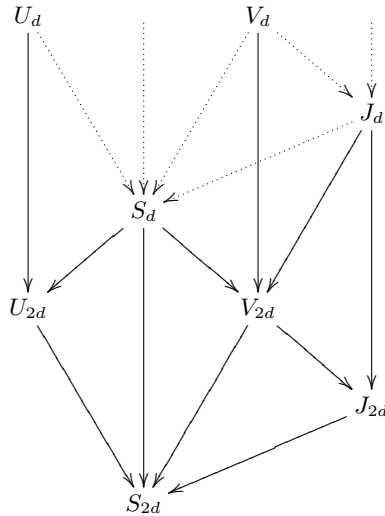

FIGURE 2. Computation of U_{2d}, V_{2d}, J_{2d} and S_{2d}

These formulas allow to efficiently compute S_{2d} from S_d and other known quantities.

- From the inverse of $S'_{d/2}$ modulo $x^{d/2}$, denoted by U_d, we use a classical Newton iteration to compute U_{2d}. Since $S_d = S_{d/2} \bmod x^{d/2+1}$, we have $U_{2d} = U_d \bmod x^{d/2}$ and we compute the coefficients of U_{2d} thanks to

$$U_{2d} = U_d \cdot \left(2 - S'_d \cdot U_d\right) \bmod x^d.$$

- From $\sqrt{H \circ S_{d/2}}$ modulo $x^{d/2}$, denoted by V_d, and the inverse of V_d modulo $x^{d/2}$, denoted by J_d, we compute V_{2d} and J_{2d} as follows. Getting V_{2d} consists in computing a solution of $v^2 - (H \circ S_d)(x) = 0$. We use

$$V_{2d} = \frac{1}{2}\left(V_d + \frac{H \circ S_d}{V_d}\right) \bmod x^d .$$

J_d and V_d are by definition inverses of each other modulo $x^{d/2}$. We obtain the inverse W_{2d} of V_d modulo x^d thanks to Newton formulas too,

$$W_{2d} = J_d \cdot (2 - V_d \cdot J_d) \bmod x^d .$$

If we now plug this value in the V_{2d} formula, we finally find

$$2\,V_{2d} = V_d + J_d \cdot (H \circ S_d) \cdot (2 - V_d \cdot J_d) \bmod x^d .$$

Another use of Newton's inversion formula yields J_{2d},

$$J_{2d} = J_d \cdot (2 - J_d \cdot V_{2d}) \bmod x^d .$$

Thanks to all these equations, we can compute U_{2d}, V_{2d}, J_{2d} from U_d, V_d, J_d and S_d. The quantity S_{2d} is then obtained from Eq. (A.2),

$$S_{2d} = S_d + \frac{V_{2d}}{2}\int U_{2d} \cdot J_{2d} \cdot \left(G \cdot (H \circ S_d) - S_d'^2\right) dx \bmod x^{2d+1} .$$

We illustrate the corresponding computations in Fig. 2.

It remains to obtain initial values, for $d = 2$. Let γ be defined by $S_2(x) = \alpha + \beta\,x + \gamma\,x^2 \bmod x^3$. The series S_2 is solution of the differential equation modulo x^2 and thus $\beta^2 + 4\,\beta\,\gamma\,x = G(x)\,H(\alpha + \beta x) \bmod x^2$. Once derivated, and evaluated at $x = 0$, we obtain γ, and thus the value of S_2,

$$S_2(x) = \alpha + \beta\,x + \left(\frac{G'(0)}{4\beta} + \frac{\beta^2\,H'(\alpha)}{4}\right) x^2 \bmod x^3 .$$

We deduce as well

$$U_2(x) = \frac{1}{\beta} \bmod x , \quad V_2(x) = 1 \bmod x \quad \text{and} \quad J_2(x) = 1 \bmod x.$$

References

[1] E.R. BERLEKAMP, *Algebraic Coding Theory*. McGraw-Hill, 1968.

[2] A. BOSTAN, F. MORAIN, B. SALVY, É. SCHOST, *Fast algorithms for computing isogenies between elliptic curves*. Mathematics of Computation **77(263)** (July 2008), 1755–1778.

[3] R.P. BRENT, F.G. GUSTAVSON, D.Y.Y. YUN, *Fast solution of Toeplitz systems of equations and computation of Padé approximants*. Journal of Algorithms **1** (1980), 259–295.

[4] H. COHEN *On the coefficients of the transformation polynomials for the elliptic modular function*. Math. Proc. Cambridge Philos. Soc. **95** (1984), 389–402.

[5] J.-M. COUVEIGNES, *Computing l-isogenies with the p-torsion*. In Proc. of the 2nd Algorithmic Number Theory Symposium (ANTS-II), volume **1122**, pages 59–65, 1996.

[6] J.-M. COUVEIGNES, R. LERCIER, *Elliptic periods for finite fields*. arXiv:0802.0165, 2008. To appear in Finite Fields and their Applications.

[7] J.-M. Couveignes, R. Lercier, *Galois invariant smoothness basis*. Series on number theory and its application **5** (2008), 154–179.

[8] J.-L. Dornstetter, *On the equivalence between Berlekamp's and Euclid's algorithms*. IEEE Transactions on Information Theory **33(3)** (1987), 428–431.

[9] A. Enge, *Computing modular polynomials in quasi-linear time*. arXiv:0704.3177, 2007. To appear in Mathematics of Computation.

[10] S.D. Galbraith, F. Hess, N.P. Smart, *Extending the GHS Weil Descent Attack*. In Proc. of Advances in Cryptology – Eurocrypt'2002, volume **2332**, pages 29–44, 2002.

[11] A. Joux, R. Lercier, *Counting points on elliptic curves in medium characteristic*. Cryptology ePrint Archive 2006/176, 2006.

[12] R. Lercier, *Computing isogenies in $GF(2^n)$*. In H. Cohen, editor, Algorithmic Number Theory: Second International Symposium, ANTS-II, volume **1122** of *LNCS*, pages 197–212. Springer Berlin / Heidelberg, May 1996.

[13] R. Lidl, H. Niederreiter, *Finite Fields*, volume **20** of Encyclopedia of Mathematics and its Applications. Addison-Wesley, 1983.

[14] J.L. Massey, *Shift-register synthesis and BCH decoding*. IEEE Transactions on Information Theory **15(1)** (1969), 122–127.

[15] V.Y. Pan, *New techniques for the computation of linear recurrence coefficients*. Finite Fields and Their Applications **6(1)** (2000), 93–118.

[16] R. Schoof, *Counting points on elliptic curves over finite fields*. Journal de théorie des nombres de Bordeaux **7(1)** (1995), 219–254.

[17] V. Shoup, *Removing Randomness from Computational Number Theory*. PhD thesis, University of Winsconsin, 1989.

[18] J.H. Silverman *The arithmetic of elliptic curves*, volume **106** of *Graduate Texts in Mathematics*. Springer-Verlag, 1986. Corrected reprint of the 1986 original.

[19] N.P. Smart, *An Analysis of Goubin's Refined Power Analysis Attack*. In Proc. of the 5th Workshop on Cryptographic Hardware and Embedded Systems (CHES'2003), volume **2779**, pages 281–290, 2003.

[20] J. Vélu, *Isogénies entre courbes elliptiques*. Comptes Rendus de l'Académie des Sciences de Paris **273** (1971), 238–241. Série A.

Reynald Lercier
DGA/CÉLAR   IRMAR
La Roche Marguerite   and   Université de Rennes 1
F-35174 Bruz   Campus de Beaulieu
                                  F-35042 Rennes
E-mail: `reynald.lercier@m4x.org`
URL: `http://perso.univ-rennes1.fr/reynald.lercier/`

Thomas Sirvent
DGA/CÉLAR   IRMAR
La Roche Marguerite   and   Université de Rennes 1
F-35174 Bruz   Campus de Beaulieu
                                  F-35042 Rennes
E-mail: `thomas.sirvent@m4x.org`
URL: `http://perso.univ-rennes1.fr/thomas.sirvent/`