Using the Charlap-Coley-Robbins polynomials for computing isogenies

François Morain

USING THE CHARLAP-COLEY-ROBBINS POLYNOMIALS FOR COMPUTING ISOGENIES

FRANÇOIS MORAIN

Abstract. The SEA algorithm for computing the cardinality of elliptic curves over finite fields in many characteristic uses modular polynomials. These polynomials come into different flavors, and methods to compute them flourished. Once equipped with some modular polynomials for prime ℓ, algebraic formulas are used to compute a curve $\mathcal{E}^*/\mathbb{F}_q$ that is ℓ-isogenous to the curve of interest $\mathcal{E}$. These formulas involve derivatives of the modular polynomial that may sometime vanish. One way to overcome this problem is to use alternative trivariate polynomials U_ℓ, V_ℓ, and W_ℓ introduced by Charlap, Coley and Robbins to overcome some difficulties in the first versions of Elkies's approach. We give properties of these polynomials, as well as formulas to compute the isogenous curve that were sketched by Atkin. Also we investigate another suggestion of Atkin using modular polynomials associated to a power product of Dedekind's η function.

1. Introduction

Computing isogenies is the central ingredient of the Schoof-Elkies-Atkin (SEA) algorithm that computes the cardinality of elliptic curves over finite fields of large characteristic [20, 2, 10] and also [4]. More recently, it is used in post-quantum cryptography [8, 13, 6, 12] among others, as well as the cryptosystems [9, 19, 11].

Given a curve $\mathcal{E}/\mathbb{F}_q$, we need to compute ℓ-isogenous curves for (small) prime ℓ's, starting from a root of some degree $\ell+1$ modular polynomial $\Psi_\ell(X,Y)$ in $X = j(\mathcal{E})$. The coefficients of the isogenous curve $\mathcal{E}^*/\mathbb{F}_q$ together with the kernel polynomial of the isogeny are computed from partial derivatives of $\Psi_\ell(X,Y)$. This method works except in cases where one of the derivatives is zero. This is a theoretical as well as practical problem. An alternative to this is to use the original approach of Elkies [10], namely using algebraic relations for A^* and B^*. Another choice is to use the CCR polynomials (U_ℓ, V_ℓ, W_ℓ) introduced in [7], for the price of finding the roots of three degree $\ell+1$ polynomials instead of 1. Several methods for computing these polynomials, as well as rational representations of A^* and B^* are given in [15] (as well as the cited references in this article).

Date: March 1, 2023.

The author is on leave from Délégation Générale pour l'Armement.

The aim of our work (which is closely related to [15]) is to give formulas to compute the necessary parameters for Elkies's algorithms, using partial derivatives of U_ℓ, in the spirit of Atkin and following the suggestion in [2]. The same work is done for the alternate polynomial U^a suggested by Atkin when $\ell \equiv 11 \bmod 12$.

The content is as follows. We review the SEA algorithm in Section 2, including formulas for Eisenstein series and introduce the CCR polynomials in Section 3. Section 4 is the central part of the article, working out the formulas giving the coefficients of the isogenous curve.

2. Schoof/Elkies/Atkin

2.1. **Prerequisites.**

2.1.1. *Division polynomials.* For $\mathcal{E} : y^2 = x^3 + Ax + B$, multiplication of a point (X, Y) by positive n on $\mathcal{E}$ is given by

$$[n](X,Y) = \left(\frac{\phi_n(X,Y)}{\psi_n(X,Y)^2}, \frac{\omega_n(X,Y)}{\psi_n(X,Y)^3}\right)$$

where the polynomials satisfy

$$\phi_n = X\psi_n^2 - \psi_{n+1}\psi_{n-1}, \quad 4Y\omega_n = \psi_{n+2}\psi_{n-1}^2 - \psi_{n-2}\psi_{n+1}^2$$

and $\phi_n, \psi_{2n+1}, \psi_{2n}/(2Y), \omega_{2n+1}/Y, \omega_{2n}$ belong to $\mathbb{Z}[A, B, X]$. It is customary to simplify this using

$$f_n(X) = \begin{cases} \psi_n(X,Y) & \text{for } n \text{ odd} \\ \psi_n(X,Y)/(2Y) & \text{for } n \text{ even} \end{cases}$$

with first values

$$f_{-1} = -1, \quad f_0 = 0, \quad f_1 = 1, \quad f_2 = 1,$$
$$f_3(X,Y) = 3X^4 + 6AX^2 + 12BX - A^2$$
$$f_4(X,Y) = X^6 + 5AX^4 + 20BX^3 - 5A^2X^2 - 4ABX - 8B^2 - A^3.$$

The degree of f_n is $(n^2-1)/2$ for odd n and $(n^2-4)/2$ for even n. If X has weight 1, A weight 2 and B weight 3, all monomials in f_n have the same weighted degree equal to the degree of f_n.

2.1.2. *Modular functions and such.* Letting $q = \exp(2i\pi\tau)$, one defines

$$E_2(q) = 1 - 24\sum_{n=1}^{\infty} \delta_1(n) q^n,$$
$$E_4(q) = 1 + 240\sum_{n=1}^{\infty} \delta_3(n) q^n,$$
$$E_6(q) = 1 - 504\sum_{n=1}^{\infty} \delta_5(n) q^n,$$

where $\delta_r(n)$ denotes the sum of the r-th powers of the divisors of n. The series E_4 and E_6 are modular forms of weight 4 and 6 respectively. The

function E_2 is not a modular form, but note that $\mathcal{F}_n(\tau) = E_2(\tau) - nE_2(n\tau)$ is a modular form of weight 2 for $\Gamma_0(n)$ and trivial multiplier (see [18] and [3]).

When $F(q) = \sum_{n \geq n_0} a_n q^n$, we introduce the operator

$$F'(q) = \frac{1}{2i\pi} \frac{dF}{d\tau} = q\frac{dF}{dq} = \sum_{n \geq n_0} n a_n q^n. \tag{1}$$

Several identities are classical:

$$\Delta = \frac{E_4^3 - E_6^2}{1728}, \quad j = \frac{E_4^3}{\Delta}, \quad j - 1728 = \frac{E_6^2}{\Delta}, \tag{2}$$

$$\frac{j'}{j} = -\frac{E_6}{E_4}, \quad \frac{j'}{j - 1728} = -\frac{E_4^2}{E_6}, \quad j' = -\frac{E_4^2 E_6}{\Delta}, \quad \frac{\Delta'}{\Delta} = E_2, \tag{3}$$

to which we add the Ramanujan differential system:

$$3{E_4}' = E_4E_2 - E_6, \quad 2{E_6}' = E_6E_2 - E_4^2, \quad 12{E_2}' = E_2^2 - E_4. \tag{4}$$

2.2. **Schoof's approach.** Let $\mathcal{E}/\mathbb{F}_q$ be an elliptic curve of cardinality $m = p + 1 - t$ with $|t| \leq 2\sqrt{q}$ by Hasse's theorem. Schoof gave the first deterministic polynomial time algorithm to compute m. The idea is to use the action of the Frobenius of $\mathcal{E}$ on ℓ-division points to find t via the characteristic equation $\phi^2 - [t]\phi + [q] = 0$ modulo (f_ℓ, ℓ).

2.3. **Using isogenies.** Elkies and Atkin gave subsequent improvements to make Schoof's algorithm efficient (and probabilistic) and usable in practice. Elkies described how to use isogenies to find factors of small degree of $f_\ell(X)$ over a finite field, provided the Frobenius equation splits modulo ℓ. Using modular polynomials, Elkies worked out a procedure to compute all the parameters needed to build a degree ℓ isogeny from $\mathcal{E} : Y^2 = X^3 + AX + B$ to some curve $\mathcal{E}^* : Y^2 = X^3 + A^*X + B^*$ and the kernel polynomial of the isogeny, thereby giving the factor we need. Atkin designed his own route towards the same goal, putting the emphasis on the use of more modular equations for $X_0(\ell)$ and its quotients.

One has (after renormalization):

$$A = -3E_4(q), B = -2E_6(q).$$

With a compatible scaling, we get

$$A^* = -3\ell^4 E_4(q^\ell), \quad B^* = -2\ell^6 E_6(q^\ell).$$

More importantly, writing σ_r for the power sums of the roots of U_ℓ, we have

$$\sigma_1 = \frac{\ell}{2}(\ell E_2(q^\ell) - E_2(q)) = -\frac{\ell}{2}\mathcal{F}_\ell(q).$$

Beyond this, Elkies proved that

$$A - A^* = 5(6\sigma_2 + 2A\sigma_0),$$
$$B - B^* = 7(10\sigma_3 + 6A\sigma_1 + 4B\sigma_0),$$

together with an induction relation satisfied by other σ_k for $k > 3$. As a consequence A^* and B^* belong to $\mathbb{Q}[\sigma_1, A, B]$ since σ_2 and σ_3 do.

Given these quantities, there are several algorithms to get the isogeny. We refer to [5] for this.

3. The polynomials of Charlap-Coley-Robbins

3.1. **Theory.** We start from an elliptic curve $\mathcal{E} : Y^2 = X^3 + AX + B$ and we fix some odd prime ℓ, putting $d = (\ell - 1)/2$. Our aim is to find the equation of an ℓ-isogenous curve $\mathcal{E}^* : Y^2 = X^3 + A^*X + B^*$.

Theorem 3.1. *There exist three polynomials U_ℓ, V_ℓ, W_ℓ in $\mathbb{Z}[X, Y, Z, 1/\ell]$ of degree $\ell+1$ in X such that $U_\ell(\sigma_1, A, B) = 0$, respectively $V_\ell(A^*, A, B) = 0$, $W_\ell(B^*, A, B) = 0$.*

Let us turn our attention to the properties of these polynomials.

Theorem 3.2. *When $\ell > 3$, the polynomials U_ℓ, V_ℓ, W_ℓ live in $\mathbb{Z}[X, Y, Z]$.*

Proposition 3.3. *Assigning respective weights 1, 2, 3 to X, Y, Z, the monomials in U_ℓ, V_ℓ and W_ℓ have generalized degree $\ell + 1$.*

3.2. **Computing isogenous curves over finite fields.** When using U_ℓ, V_ℓ, W_ℓ, we need to find the roots of three polynomials of degree $\ell+1$ instead of 1. In general, if U_ℓ has rational roots (it should be 1, 2 or $\ell + 1$), then this is the case for each of V_ℓ, W_ℓ. For each triplet of solutions (σ_1, z_1, z_2) we need to test whether this leads to an isogeny or not. To speed up things, we may compute rational fractions for A^* and B^* as explained in [16] (see also [17, §7]). Another path was sketched by Atkin in [2], and this is what we describe next.

4. Revisiting CCR *à la* Atkin

The idea is to generalize the approach in [1, 2, 14], that is exploit q-series identities to get the parameters (σ, A^*, B^*), where we write σ for σ_1 from now on.

4.1. **Properties of U_ℓ.** We write for readability $U = U_\ell$ and

$$\partial_\sigma = \frac{\partial U}{\partial \sigma}, \partial_4 = \frac{\partial U}{\partial E_4}, \partial_6 = \frac{\partial U}{\partial E_6}.$$

and propagate the notation to double derivatives.

The polynomial U is homogeneous with weights, so that

$$(\ell + 1)U = \sigma\partial_\sigma + 2E_4\partial_4 + 3E_6\partial_6. \tag{5}$$

Note that partial derivatives of U are also homogeneous polynomials and we find

$$\ell\partial_\sigma = \sigma\partial_{\sigma\sigma} + 2E_4\partial_{\sigma 4} + 3E_6\partial_{\sigma 6}, \tag{6}$$

$$(\ell - 1)\partial_4 = \sigma\partial_{\sigma 4} + 2E_4\partial_{44} + 3E_6\partial_{46}, \tag{7}$$

$$(\ell - 2)\partial_6 = \sigma\partial_{\sigma 6} + 2E_4\partial_{46} + 3E_6\partial_{66}. \tag{8}$$

4.2. **Getting the isogenous curve from CCR polynomials.**

4.2.1. *Finding $\tilde{E}_4$.*

Proposition 4.1. *The value of $\tilde{E}_4$ is given by*

$$-\frac{4\ell(3E_4^2\partial_6 + 2E_6\partial_4) - \partial_\sigma(\ell^2 E_4 + 4\sigma^2)}{\ell^4\partial_\sigma}.$$

Proof: We differentiate (using (1)) $U(\sigma, E_4, E_6) = 0$ to get

$$\sigma'\partial_\sigma + E_4'\partial_4 + E_6'\partial_6 = 0. \tag{9}$$

We differentiate $\sigma = \ell/2(\ell\tilde{E}_2 - E_2)$ leading to

$$\sigma' = \frac{\ell}{2}\,(\ell^2\tilde{E}_2' - E_2') = \frac{\ell}{24}\,(\ell^2(\tilde{E}_2^2 - \tilde{E}_4) - (E_2^2 - E_4)).$$

Replace $\ell\tilde{E}_2$ by $2\sigma/\ell + E_2$ to get

$$\sigma' = \frac{\ell}{24}\,\left(\frac{4\sigma^2}{\ell^2} + \frac{4\sigma}{\ell}E_2 - (\ell^2\tilde{E}_4 - E_4)\right),$$

that we plug in (9) together with the expressions for ${E_4}'$ and ${E_6}'$ from equation (4) to get a polynomial of degree 1 in E_2 whose coefficient of E_2 is

$$\sigma\partial_\sigma + 2E_4\partial_4 + 3E_6\partial_6,$$

which we recognize in (5). Therefore, we get

$$(\ell+1)UE_2 + \frac{\ell\,\partial_\sigma}{4}\left(\frac{4\sigma^2}{\ell^2} - (\ell^2\tilde{E}_4 - E_4)\right) - 2E_6\partial_4 - 3E_4^2\partial_6 = 0 \tag{10}$$

from which we deduce $\tilde{E}_4$ since $U(\sigma, E_4, E_6) = 0$. □

4.2.2. *Finding $\tilde{E}_6$.*

Proposition 4.2. *The value of $\tilde{E}_6$ may be written*

$$\tilde{E}_6 = -\frac{N}{\ell^6\,\partial_\sigma^3}$$

where N is some polynomial of degree 3 in ℓ and given at the end of the proof.

Proof: We differentiate (9).

$$\sigma''\partial_\sigma + \sigma'(\sigma'\partial_{\sigma\sigma} + E_4'\partial_{\sigma 4} + E_6'\partial_{\sigma 6}) \tag{11}$$

$$+\quad E_4''\partial_4 + E_4'(\sigma'\partial_{4\sigma} + E_4'\partial_{44} + E_6'\partial_{46}) \tag{12}$$

$$+\quad E_6''\partial_6 + E_6'(\sigma'\partial_{6\sigma} + E_4'\partial_{64} + E_6'\partial_{66}) = 0 \tag{13}$$

We compute in sequence

$$12E_2'' = 2E_2E_2' - E_4' = E_2(E_2^2 - E_4)/6 - (E_2E_4 - E_6)/3,$$

$$12\tilde{E}_2'' = 2\tilde{E}_2\tilde{E}_2' - \tilde{E}_4' = \tilde{E}_2(\tilde{E}_2^2 - \tilde{E}_4)/6 - (\tilde{E}_2\tilde{E}_4 - \tilde{E}_6)/3,$$

which give us the value

$$\sigma'' = \frac{\ell}{2}\,(\ell^3\tilde{E}_2'' - E_2'')$$

to be used in (11). Differentiating relations of (4), we get

$$E_4'' = \frac{1}{3}\,(E_2'E_4 + E_2E_4' - E_6'), \quad E_6'' = \frac{1}{2}\,(E_2'E_6 + E_2E_6' - 2E_4E_4'),$$

to be used in lines (12) and (13) respectively. We replace $\tilde{E}_4$ by its value from (10), and $\tilde{E}_2$ using $\sigma = (\ell/2)(\ell\tilde{E}_2 - E_2)$. This finally yields an expression as polynomial in E_2:

$$C_2E_2^2 + C_1E_2 + C_0 = 0.$$

The unknown $\tilde{E}_6$ is to be found in C_0 only.

By luck(?)

Proposition 4.3. *The coefficients C_1 and C_2 vanish for a triplet such that $U_\ell(\sigma, E_4, E_6) = 0$.*

Sketch of the proof: The strategy to prove this is the same in both cases. Replace $\partial_{\sigma\sigma}$, ∂_{44} and ∂_{66} by their values from (6). Factoring the resulting expressions yields the same factor $\sigma\partial_\sigma + 2E_4\partial_4 + 3E_6\partial_6$, which cancels C_1 and C_2. We add a `SageMath` script for the convenience of the reader as an appendix to this work. □

We are left with

$$\tilde{E}_6 = -\frac{N}{\ell^6\,\partial_\sigma^3}$$

where N is a polynomial in degree 3 in ℓ

$$N = -E_6\partial_\sigma^3\ell^3 + c_2\ell^2 + 12\partial_\sigma^2\sigma(3E_4^2\partial_6 + 2E_6\partial_4)\ell - \partial_\sigma^3\sigma^3.$$

The coefficient c_2 is heavy looking and we give slightly factored as a polynomial in E_4:

$$\begin{aligned} c_2 \;=\;& 18(\partial_6^2\partial_{\sigma\sigma} - 2\partial_6\partial_\sigma\partial_{\sigma 6} + \partial_{66}\partial_\sigma^2)E_4^4 \\ &+(24E_6\partial_4(\partial_6\partial_{\sigma\sigma} - \partial_\sigma\partial_{\sigma 6}) + 24E_6\partial_\sigma(\partial_{46}\partial_\sigma - \partial_6\partial_{\sigma 4}) + 10\partial_4\partial_\sigma^2)E_4^2 \\ &+3\partial_\sigma^2(7E_6\partial_6 - \sigma\partial_\sigma)E_4 + 8E_6^2(\partial_4^2\partial_{\sigma\sigma} - 2\partial_4\partial_\sigma\partial_{\sigma 4} + \partial_{44}\partial_\sigma^2).\square \end{aligned}$$

4.2.3. *Numerical example.* Consider $E : Y^2 = X^3 + X + 3$ over $\mathbb{F}_{1009}$ and $\ell = 5$. Using

$$U_5(X) = X^6 + 20X^4A + 160X^3B - 80X^2A^2 - 128XAB - 80B^2,$$

we select $\sigma = 584$ and compute

$$\partial_\sigma = 905, \partial_4 = 779, \partial_6 = 140$$

from which $\tilde{E}_4 = 497$, $A^* = 441$. After tedious computations, we find $B^* = 997$.

4.3. **The case $\ell \equiv 11 \bmod 12$.** In this case, Atkin suggests to replace σ with $f(q) = (\eta(q)\eta(q^\ell))^2$, where η is Dedekind's function. The corresponding modular polynomial U_ℓ^a can be computed using the techniques described in [15]. For instance (using the basis with E_4, E_6 and Δ):

$$U_{11}^a(X) = X^{12} - 990\,\Delta\,X^6 + 440\,E_4\,\Delta\,X^4 - 165\,E_6\,\Delta\,X^3 + 22\,E_4^2\,\Delta X^2 - E_6\,E_4\,\Delta\,X - 11\,\Delta^2,$$

which is sparser than $U_{11}(X)$.

4.3.1. *Some properties of U_ℓ^a.* Let v_2 (resp. v_3) be the maximal power of $1/2$ (resp. $1/3$) of the coefficients of $U_\ell^a(X)$ that we found experimentally

ℓ	v_2	v_3
11	16	12
23	32	24
47	64	48
59	80	60
71	96	72

It seems that $f/12$ should be a more sensible choice, leading to $12^{\ell+1}U_\ell^a(X/12)$ having integer coefficients.

We have formulas analogous to (6), due the corresponding homogeneous property

$$\ell\partial_f = f\partial_{ff} + 2E_4\partial_{f4} + 3E_6\partial_{f6}, \tag{14}$$

$$(\ell-1)\partial_4 = f\partial_{f4} + 2E_4\partial_{44} + 3E_6\partial_{46}, \tag{15}$$

$$(\ell-2)\partial_6 = f\partial_{f6} + 2E_4\partial_{46} + 3E_6\partial_{66}. \tag{16}$$

4.3.2. *Computing σ, $\tilde{E}_4$ and $\tilde{E}_6$.*

Proposition 4.4. *The value of σ is*

$$\sigma = \frac{\ell\,\left(3\,\partial_6\,E_4^2 + 2\,\partial_4\,E_6\right)}{f\partial_f}.$$

Proof: Remark that $f^{12} = \Delta(z)\Delta(\ell z)$ and therefore we deduce the discriminant $\tilde{\Delta} = f^{12}/\Delta$ of the isogenous curve. We have also (using (3)):

$$12\,\frac{f'}{f} = \frac{\Delta'}{\Delta} + \ell\frac{\tilde{\Delta}'}{\tilde{\Delta}} = E_2 + \ell\tilde{E}_2,$$

from which we deduce f'. Again, U_ℓ^a is homogeneous with weight $\ell+1$, so that we have identities similar to those for σ. In particular

$$(\ell+1)U_\ell^a = f\partial_f + 2E_4\partial_4 + 3E_6\partial_6. \tag{17}$$

Starting from $f'\partial_f + E_4'\partial_4 + E_6'\partial_6 = 0$, and replacing by the known values, we find

$$(f\partial_f + 4E_4\partial_4 + 6E_6\partial_6)\,E_2 + f\ell\tilde{E}_2\partial_f - 6E_4^2\partial_6 - 4E_6\partial_4 = 0,$$

which is

$$f\partial_f(\ell\tilde{E}_2 - E_2) - 6E_4^2\partial_6 - 4E_6\partial_4 = 0,$$

and this gives us the result. □

Proposition 4.5. *The value of $\tilde{E}_4$ is given by*

$$\tilde{E}_4 = -\frac{M}{\ell^2 f^2 E_4 E_6 \partial_f^3}$$

where M is a polynomial given at the end of the proof.

Proof: We differentiate f' to obtain:

$$\begin{aligned} f'' &= \frac{1}{12}\left(f'(\ell\tilde{E}_2 + E_2) + f(\ell^2\tilde{E}_2' + E_2')\right) \\ &= \frac{f}{12^2}\left((\ell\tilde{E}_2 + E_2)^2 + \ell^2(\tilde{E}_2^2 - \tilde{E}_4) + (E_2^2 - E_4))\right). \end{aligned}$$

We inject this together and the diagonal derivatives of (14) and $\tilde{E}_2 = (E_2 + 2\sigma/\ell)/\ell$ into

$$f''\partial_f + f'(f'\partial_{ff} + E_4'\partial_{f4} + E_6'\partial_{f6}) \tag{18}$$

$$+ \quad E_4''\partial_4 + E_4'(f'\partial_{4f} + E_4'\partial_{44} + E_6'\partial_{46}) \tag{19}$$

$$+ \quad E_6''\partial_6 + E_6'(f'\partial_{6f} + E_4'\partial_{64} + E_6'\partial_{66}) = 0 \tag{20}$$

to get a polynomial of degree 2 in E_2 whose coefficients of degree 2 and 1 turn out to vanish. We are left with

$$\tilde{E}_4 = -\frac{M}{\ell^2 f^2 E_4 E_6 \partial_f^3}$$

where

$$\begin{aligned} M &= 24(3E_6\partial_6^2\partial_{f4} + \partial_{46}\partial_f^2 f)E_4^6 \\ &\quad +12(9E_6^2\partial_6^2\partial_{f6} - 3E_6\partial_6^2\partial_f\ell + 6E_6\partial_6\partial_f\partial_{f6}f - \partial_6\partial_f^2\ell f + \partial_f^2\partial_{f6}f^2 \\ &\qquad -6E_6\partial_6^2\partial_f + 2\partial_6\partial_f^2 f)E_4^5 + 96E_4^4E_6^2\partial_4\partial_6\partial_{f4} \\ &\quad +4E_6(36E_6^2\partial_4\partial_6\partial_{f6} - 12E_6\partial_4\partial_6\partial_f\ell + 12E_6\partial_4\partial_f\partial_{f6}f - 12E_6\partial_{46}\partial_f^2 f \\ &\qquad +12E_6\partial_6\partial_f\partial_{f4}f - 24E_6\partial_4\partial_6\partial_f - 5\partial_4\partial_f^2 f)E_4^3 \\ &\quad +E_6(32E_6^2\partial_4^2\partial_{f4} - 42E_6\partial_6\partial_f^2 f + \partial_f^3 f^2)E_4^2 \\ &\quad +16E_6^3\partial_4(3E_6\partial_4\partial_{f6} - \partial_4\partial_f\ell + 2\partial_f\partial_{f4}f - 2\partial_4\partial_f)E_4 \\ &\quad +24E_6^4\partial_{46}f\partial_f^2 - 8E_6^3\partial_4\ell f\partial_f^2 + 8E_6^3\partial_{f4}f^2\partial_f^2 + 8E_6^3\partial_4 f\partial_f^2. \quad \square \end{aligned}$$

Finally, we remark that B^* satisfies

$$B^{*2} + 6912\tilde{\Delta} - 4A^{*3}/27 = 0, \quad U_\ell^a(-\ell f, A^*, B^*) = 0, \tag{21}$$

the latter relation coming from applying the Atkin-Lehner involution to the modular form for σ_1. The gcd of these two polynomials should reveal B^*. In the rare case where this gcd has degree 2 (which would imply two elliptic

curves being isogenous to E), we would be forced to use higher differentials, which would look like a formidable task.

4.3.3. *Numerical example.* Consider $E : Y^2 = X^3 + X + 3$ over $\mathbb{F}_{1009}$. The polynomial U_{11}^a has two roots: 65 and 333. We take $f = 65$. We first compute $\sigma = 75$. Then $\tilde{E}_4 = 532$. The gcd of the two polynomials in (21) has degree 1 and root $B^* = 460$.

5. Conclusions

We have completed the task suggested by Atkin for using the CCR polynomials in building isogenies. All these formulas require $O(\ell^2)$ multiplications in the base field, due to the computation of partial derivatives of polynomials of degree $O(\ell)$. Note that this is the same cost as using the rational fractions giving A^* and B^*, but less storage is needed.

As a consequence, we have several algorithms and formulas to be used, depending on the practical problem to be solved.

References

[1] A. O. L. Atkin. The number of points on an elliptic curve modulo a prime. Draft, 1988.

[2] A. O. L. Atkin. The number of points on an elliptic curve modulo a prime (II). Draft. Available on `http://listserv.nodak.edu/archives/nmbrthry.html`, 1992.

[3] Bruce C. Berndt. Ramanujan's formulas for Eisenstein series. In *Number theory and related topics (Bombay, 1988)*, volume 12 of *Tata Inst. Fund. Res. Stud. Math.*, pages 23–29. Tata Inst. Fund. Res., Bombay, 1989.

[4] I. Blake, G. Seroussi, and N. Smart. *Elliptic curves in cryptography*, volume 265 of *London Math. Soc. Lecture Note Ser.* Cambridge University Press, 1999.

[5] A. Bostan, F. Morain, B. Salvy, and É. Schost. Fast algorithms for computing isogenies between elliptic curves. *Math. Comp.*, 77(263):1755–1778, 2008.

[6] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: an efficient post-quantum commutative group action. In Thomas Peyrin and Steven D. Galbraith, editors, *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part III*, volume 11274 of *Lecture Notes in Computer Science*, pages 395–427. Springer, 2018.

[7] L. S. Charlap, R. Coley, and D. P. Robbins. Enumeration of rational points on elliptic curves over finite fields. Draft; a copy is available at `http://www.lix.polytechnique.fr/Labo/Francois.Morain/Introuvables/Drafts/ccr.pdf`, 1991.

[8] Denis Xavier Charles, Kristin E. Lauter, and Eyal Z. Goren. Cryptographic hash functions from expander graphs. *J. Cryptol.*, 22(1):93–113, 2009.

[9] Jean-Marc Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291, 2006. `http://eprint.iacr.org/2006/291`.

[10] N. D. Elkies. Elliptic and modular curves over finite fields and related computational issues. In D. A. Buell and J. T. Teitelbaum, editors, *Computational Perspectives on Number Theory: Proceedings of a Conference in Honor of A. O. L. Atkin*, volume 7 of *AMS/IP Studies in Advanced Mathematics*, pages 21–76. American Mathematical Society, International Press, 1998.

[11] Luca De Feo, Jean Kieffer, and Benjamin Smith. Towards practical key exchange from ordinary isogeny graphs. In Thomas Peyrin and Steven D. Galbraith, editors, *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part III*, volume 11274 of *Lecture Notes in Computer Science*, pages 365–394. Springer, 2018.

[12] Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. Sqisign: Compact post-quantum signatures from quaternions and isogenies. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part I*, volume 12491 of *Lecture Notes in Computer Science*, pages 64–93. Springer, 2020.

[13] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29 - December 2, 2011. Proceedings*, volume 7071 of *Lecture Notes in Computer Science*, pages 19–34. Springer, 2011.

[14] F. Morain. Calcul du nombre de points sur une courbe elliptique dans un corps fini : aspects algorithmiques. *J. Théor. Nombres Bordeaux*, 7:255–282, 1995.

[15] François Morain. Computing the Charlap-Coley-Robbins modular polynomials, 2023.

[16] Masayuki Noro, Masaya Yasuda, and Kazuhiro Yokoyama. Symbolic computation of isogenies of elliptic curves by Vélu's formula. *Comment. Math. Univ. St. Pauli*, 68:93–130, 2020.

[17] Adrien Poteaux and Éric Schost. Modular composition modulo triangular sets and applications. *Comput. Complexity*, 22(3):463–516, 2013.

[18] S. Ramanujan. Modular equations and approximations to π. *Quarterly J. Math.*, XLV:350–372, 1914.

[19] A. Rostovtsev and A. Stolbunov. Public-key cryptosystem based on isogenies. Cryptology ePrint Archive, Report 2006/145, 2006. `http://eprint.iacr.org/`.

[20] R. Schoof. Counting points on elliptic curves over finite fields. *J. Théor. Nombres Bordeaux*, 7:219–254, 1995.

[21] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.2)*, 2020. `https://www.sagemath.org`.

Appendix A. A script to check the computations

This `SageMath` [21] script can also be downloaded from the author's web page.

```
# This script is devoted to the computation and verification of several
# identities related to CCR polynomials using the notations of the preprin

# one ring to rule them all
R.<ell,E2,E4,E6,sigma,E4t,E6t,d4,d6,s,ds,ds4,ds6,d46,f,df,df4,df6>
    =PolynomialRing(Rationals(),18)

########## The CCR case

# returns ell^-4 * ds^-1 * (-12*ell*E4^2*d6 + ...)
```

```
def check_E4t():
    E4p=(E2*E4 - E6)/3
    E6p=(E2*E6-E4^2)/2
    E2p=(E2^2-E4)/12
    E2t=(E2+2*sigma/ell)/ell
    sigp=ell/24*(4*sigma^2/ell^2+4*sigma/ell*E2-(ell^2*E4t-E4))
    tmp=sigp*ds+E4p*d4+E6p*d6
    tmp=tmp.numerator()
    print("degree(tmp, E2)=", tmp.degree(E2))
    # check that coeff of E2 is zero
    c1=tmp.coefficient({E2:1})
    # is a multiple of (2*E4*d4 + 3*E6*d6 + f*df), hence 0
    print("c1=", c1.factor())
    # find sigma as a root of constant coefficient
    e4t=tmp.coefficient({E2:0})
    e4t=-e4t.coefficient({E4t:0})/e4t.coefficient({E4t:1})
    # sig contains the value of sigma
    return e4t.factor()

# returns
# ell^-6 * ds^-3 * sigma^-1 * E6^-1 * E4^-1 * (-18*ell^3*E4^5*E6*d6^2*ds+.
def check_E6t():
    e4t=check_E4t()
    E4p=(E2*E4 - E6)/3
    E6p=(E2*E6-E4^2)/2
    E2p=(E2^2-E4)/12
    E2t=(E2+2*sigma/ell)/ell
    sigp=ell*(4*sigma^2/ell^2+4*sigma/ell*E2-(ell^2*e4t-E4))/24
    # more derivatives
    E4pp=1/3*(E2p*E4+E2*E4p-E6p)
    E6pp=1/2*(E2p*E6+E2*E6p-2*E4*E4p)
    # crucial values
    E4tp=1/3*(E2t*e4t-E6t)
    E2tp=(E2t^2-e4t)/12
    E2pp=1/12*(2*E2*E2p-E4p)
    E2tpp=1/12*(2*E2t*E2tp-E4tp)
    sigpp=ell*(ell^3*E2tpp-E2pp)/2
    # inject diagonal derivatives
    dss = (ell*ds     -2*E4*ds4 -3*E6*ds6)/sigma
    d44 = ((ell-1)*d4-sigma*ds4-3*E6*d46)/(2*E4)
    d66 = ((ell-2)*d6-sigma*ds6-2*E4*d46)/(3*E6)
    # starting point
```

```
    tmp=        sigpp*ds+sigp*(sigp*dss+E4p*ds4+E6p*ds6)
    tmp=tmp + E4pp*d4+E4p*(sigp*ds4+E4p*d44+E6p*d46)
    tmp=tmp + E6pp*d6+E6p*(sigp*ds6+E4p*d46+E6p*d66)
    tmp=tmp.numerator()
    c2=tmp.coefficient({E2:2})
    print("E6t.c2=", c2.factor())
    c1=tmp.coefficient({E2:1})
    print("E6t.c1=", c1)
    c0=tmp.coefficient({E2:0})
    e6t=-c0.coefficient({E6t:0})/c0.coefficient({E6t:1})
    return e6t.factor()

########## The case of ell = 11 mod 12, Atkin's variant
def check11_sigma():
#     R.<ell,E2,E4,E6,sigma,d4,d6,f,df>=PolynomialRing(Rationals(),9)
    tmp=2*E4*d4+3*E6*d6+f*df
    E4p=(E2*E4 - E6)/3
    E6p=(E2*E6-E4^2)/2
    E2p=(E2^2-E4)/12
    E2t=(E2+2*sigma/ell)/ell
    fp=f/12*(ell*E2t+E2)
    tmp=fp*df+E4p*d4+E6p*d6
    tmp=tmp.numerator()
    # check that coeff of E2 is zero
    c1=tmp.coefficient({E2:1})
    # is a multiple of (2*E4*d4 + 3*E6*d6 + f*df), hence 0
    print("c1=", c1.factor())
    # find sigma as a root of constant coefficient
    sig=tmp.coefficient({E2:0})
    sig=-sig.coefficient({sigma:0})/sig.coefficient({sigma:1})
    # sig contains the value of sigma
    return sig.factor()

# returns
# (-1) * df^-3 * f^-2 * ell^-2 * E6^-1 * E4^-1 * (-36*ell*E4^5*E6*d6^2*df+
def check11_E4t():
    sig=check11_sigma()
    E4p=(E2*E4 - E6)/3
    E6p=(E2*E6-E4^2)/2
    E2p=(E2^2-E4)/12
    E2t=(E2+2*sig/ell)/ell
    fp=f/12*(ell*E2t+E2)
```

```
fpp=f/12^2*((ell*E2t+E2)^2+ell^2*(E2t^2-E4t)+(E2^2-E4))
E4pp=1/3*(E2p*E4+E2*E4p-E6p)
E6pp=1/2*(E2p*E6+E2*E6p-2*E4*E4p)
# inject diagonal derivatives
dff = (      ell*df-2*E4*df4  -3*E6*df6)/f
d44 = ((ell-1)*d4-f*df4-3*E6*d46)/(2*E4)
d66 = ((ell-2)*d6-f*df6-2*E4*d46)/(3*E6)
tmp=         fpp*df+ fp*(fp*dff+E4p*df4+E6p*df6)
tmp=tmp + E4pp*d4+E4p*(fp*df4+E4p*d44+E6p*d46)
tmp=tmp + E6pp*d6+E6p*(fp*df6+E4p*d46+E6p*d66)
tmp=tmp.numerator()
print("degree(tmp, E2)=", tmp.degree(E2))
c2=tmp.coefficient({E2:2})
print("E4t.c2=", c2.factor())
c1=tmp.coefficient({E2:1})
print("E4t.c1=", c1)
c0=tmp.coefficient({E2:0})
e4t=-c0.coefficient({E4t:0})/c0.coefficient({E4t:1})
return e4t.factor()
```

LIX - Laboratoire d'informatique de l'École polytechnique *and* GRACE - Inria Saclay–Île-de-France
Email address: morain@lix.polytechnique.fr