USING FRICKE MODULAR POLYNOMIALS TO COMPUTE ISOGENIES

FRANÇOIS MORAIN

ABSTRACT. Let $\mathcal{E}$ be an elliptic curve over a field $\mathbf{K}$ and ℓ a prime. There exists an elliptic curve $\mathcal{E}^*$ related to $\mathcal{E}$ by an isogeny of degree ℓ only if $\Phi_\ell^t(X, j(\mathcal{E})) = 0$, where $\Phi_\ell^t(X, Y)$ is the traditional modular polynomial. Moreover, Φ_ℓ^t gives the coefficients of $\mathcal{E}^*$, together with parameters needed to build the isogeny explicitly. Since Φ_ℓ^t has very large coefficients, many families with smaller coefficients can be used instead, as described by Elkies, Atkin and others. In this work, we concentrate on the computation of the family of modular polynomials introduced by Fricke and more recently used by Charlap, Coley and Robbins. In some cases, the resulting polynomials are small, which justifies the interest of this study. We review and adapt the known algorithms to perform the computations of these polynomials. After describing the use of series computations, we investigate fast algorithms using floating point numbers based on fast numerical evaluation of Eisenstein series. We also explain how to use isogeny volcanoes as an alternative. The last part is concerned with finding explicit formulas for computing the coefficients of $\mathcal{E}^*$. To this we add tables of numerical examples.

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 [40, 2, 20] and also [6]. More recently, it has found its way in post-quantum cryptography [14, 27, 11, 25] among others, as well as the cryptosystems [17, 39, 24].

Let $\mathbf{K}$ be a field of characteristic different from 2 and 3. A (separable) isogeny between two elliptic curves $\mathcal{E}/\mathbf{K} : Y^2 = X^3 + AX + B$ and $\mathcal{E}^*/\mathbf{K} : Y^2 = X^3 + A^*X + B^*$ is a group morphism that is a rational map of degree N (the cardinality of its kernel assumed to be cyclic). There are two ways to handle these isogenies. When the degree N is small, formulas for A^*, B^* and the kernel polynomial can be precomputed. For large N, one of the key ingredients is modular polynomials, the second one finding rational expressions for A^* and B^* from A, B and the modular polynomial. Note there is a purely algebraic approach using triangular sets [36, §7] (see also [35]).

Date: February 13, 2024.

There are many families of modular polynomials that can be used, with different properties. Very generally, a modular polynomial is some bivariate polynomial $\Phi(X, J)$ where J corresponds to the j-invariant of the elliptic curve $\mathcal{E}$, and X stands for some modular function on $\Gamma_0(N)$. The prototype is $j(\mathcal{E}^*)$ (see below for more precise statements) that yields traditional modular polynomials. Alternative choices for X exist. They all lead to polynomials of (conjectured) height $O(\psi(N)\log N)$ but with small constants. Here $\psi(N) = N\prod_{p|N}(1+1/p)$ is the cardinality of $\Gamma_0(N)\backslash\Gamma$.

In [26], Fricke computes a resolvant polynomial U_N of degree $\psi(N)$ for $\wp(Nz)$ where $\wp$ is the Weierstrass $\wp$-function of $\mathcal{E}$ and N small. It turns out to be the same polynomial as used in Elkies's work and also by [12] (without notice). The latter authors complete this with two polynomials V_N, W_N having the property (among others) that A^* (resp. B^*) is a root of $V_N(X, A, B)$ (resp. $W_N(X, A, B)$). The polynomial U_N is the modular polynomial associated with a form of weight 2 for $\Gamma_0(N)$ related to the Eisenstein series E_2; V_N (resp. W_N) is the modular polynomial for $E_4(N\tau)$ (resp. $E_6(N\tau)$).

The aim of this work is to describe the properties of the polynomials (U_N, V_N, W_N) and the relevant algorithms to compute them, extending methods already used in the traditional cases of bivariate modular polynomials based on modular functions. These methods apply to modular forms of even weight and yield trivariate polynomials in (E_4, E_6, Δ), see below for the rationale. Though our primary interest is in prime values $N = \ell$, the theory and practice for the general case of an integer N follow the same paths. In Appendix B.2, we give some details on an example when $N = 6$.

Section 2) recalls results on classical functions and modular forms, adding approaches to recognize a collection of modular forms as polynomials in the traditional quantities E_4, E_6 and Δ. Section 3 addresses fast methods to evaluate these functions, including an approach to the simultaneous evaluation of several classical series. Section 4 gathers properties of Fricke polynomials; Section 5 is devoted to various algorithms for performing the computations developed for classical modular polynomials. This includes the computation of algebraic expressions for A^* and B^* as rational fractions (see [35]). Section 6 explains how to compute the isogenous curve using partial derivatives of the polynomial U_ℓ, in the spirit of Atkin's work in the traditional case. We give numerical examples and height comparisons between modular polynomials in Section 7. An appendix contains numerical values for our polynomials, as well as a script for checking the results of Section 6.

Notations: Let τ be a complex number in the upper half plane. Put $q_1 = \exp(i\pi\tau)$ and $q = q_1^2$. Depending on authors, formulas are expressed in either parameter, which sometimes is clumsy. We write indifferently $f(\tau)$ or $f(q)$ some series. We denote by n^ω the complexity of multiplying two $n \times n$

matrices over some field, that is used in the analysis of computer algebra algorithms, in our case solving linear systems.

2. Modular forms

For convenience, we follow [15] and the references given below are related to this.

2.1. **Elements of theory.**

2.1.1. *Integer matrices.* We note:

$$\Gamma = \mathrm{Sl}_2(\mathbb{Z}) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix}, ad - bc = 1 \right\}.$$

Let $n > 0$ be an integer and $\mathcal{M}_n$ be the set of primitive integral 2×2 matrices of determinant n. The relation $R \sim_\Gamma R'$ means $\Gamma R = \Gamma R'$ for R, R' in $\mathcal{M}_n$.

Proposition 2.1 (Prop. 6.5.3)**.** *The matrices*

$$R_i = \begin{pmatrix} a_i & b_i \\ 0 & d_i \end{pmatrix}, 1 \leq i \leq \psi(n)$$

with $a_i > 0$, $a_i d_i = n$, $\gcd(a_i, b_i, d_i) = 1$ and $0 \leq b_i < d_i$, form a set of representatives of $\mathcal{M}_n$ modulo $\sim_\Gamma$.

Proposition 2.2 (Prop. 6.5.3 – cont'd)**.** *Let $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ be of determinant N. Then*

$$M \sim_\Gamma R = \begin{pmatrix} A & B \\ 0 & D \end{pmatrix}$$

where $A = \gcd(a, c)$ and $D = N/A$.

Proof: We look for $V = \begin{pmatrix} u & v \\ w & x \end{pmatrix} \in \Gamma$ such that $VM = \begin{pmatrix} A & B \\ 0 & D \end{pmatrix}$ for $AD = N$, $\gcd(A, B, D) = 1$. This is equivalent to

$$ua + vc = A, \tag{1}$$
$$wa + xc = 0, \tag{2}$$
$$ub + vd = B, \tag{3}$$
$$wb + xd = D. \tag{4}$$

A natural candidate for A is $\gcd(a, c) = ua + vc$ in integers u and v. Note that $A \mid N$ since $ad - bc = N$. Write $a = Aa'$, $c = Ac'$ with $\gcd(a', c') = 1$. Equation (2) has solutions $w = -c'$ and $x = a'$, which is coherent with equation (4), with the choice of $D = N/A$. We are left with $B_0 = ub + vd = B + qD$, with $0 \leq B < D$ so that

$$\begin{pmatrix} 1 & -q \\ 0 & 1 \end{pmatrix} \cdot VM = \begin{pmatrix} 1 & -q \\ 0 & 1 \end{pmatrix} \begin{pmatrix} A & B + qD \\ 0 & D \end{pmatrix} = \begin{pmatrix} A & B \\ 0 & D \end{pmatrix}.$$

We finish with

$$U = V^{-1} \cdot \begin{pmatrix} 1 & q \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} a' & a'q - v \\ c' & c'q + u \end{pmatrix}$$

and we are done. □

We denote by

$$\Gamma_0(N) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma, c \equiv 0 \bmod N \right\},$$

and $\psi(N) = N \prod_{p|N}(1 + 1/p)$ the cardinal of $\Gamma_0(N)\backslash\Gamma$.

The general case is treated in Corollary 6.2.11 but we need the prime case only.

Proposition 2.3 (Ex. 6.2.12)**.** *Let $N = \ell$ be a prime number. A system of representative of cosets for $\Gamma_0(\ell)\backslash\Gamma$ is formed of the matrices $R_c = \begin{pmatrix} 1 & 0 \\ c & 1 \end{pmatrix}$ for $0 \leq c < \ell$, and $R_\ell = \begin{pmatrix} 0 & -1 \\ 1 & \ell \end{pmatrix}$.*

The following result will prove useful for computing conjugate values of the functions that we study later on.

Lemma 2.4. *For $0 < c < \ell$:*

$$\begin{pmatrix} \ell & 0 \\ c & 1 \end{pmatrix} = \begin{pmatrix} \ell & -\overline{c} \\ c & u \end{pmatrix} \begin{pmatrix} 1 & \overline{c} \\ 0 & \ell \end{pmatrix} = T_c \, R'_c$$

with $u\ell + \overline{c}c = 1$;

$$\begin{pmatrix} 0 & -\ell \\ 1 & \ell \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & \ell \end{pmatrix} = T_\ell \, R'_\ell.$$

2.1.2. *Modular forms.* Let f be a modular form for Γ of integer weight $2k$. By construction

$$f(M\tau) = (c\tau + d)^{2k} f(\tau)$$

for any matrix $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ in Γ.

If $M \in \mathrm{GL}_2(\mathbb{R})$, define

$$f|_{2k}M(\tau) = \det(M)^k (c\tau + d)^{-2k} f(M\tau).$$

2.2. **Eisenstein series.** The classical Eisenstein series[1] we consider are

$$E_2(q) = 1 - 24 \sum_{n=1}^{\infty} \delta_1(n) q^n,$$

$$E_4(q) = 1 + 240 \sum_{n=1}^{\infty} \delta_3(n) q^n,$$

[1]Ramanujan used $L = P = E_2$, $M = Q = E_4$, $N = R = E_6$.

$$E_6(q) = 1 - 504 \sum_{n=1}^{\infty} \delta_5(n) q^n,$$

where $\delta_r(n)$ denotes the sum of the r-th powers of the divisors of n. Other series E_{2k} can be defined for even $k > 3$. The series E_{2k} is a modular form of weight $2k$ for $k > 1$.

Also of interest is the discriminant Δ:

$$\Delta(q) = (E_4(q)^3 - E_6(q)^2)/1728 = \eta(q)^{24}.$$

Dedekind's function is $\eta(q) = q^{1/24} \prod_{n=1}^{\infty}(1 - q^n)$.

Finally, the modular invariant is

$$j(q) = \frac{E_4(q)^3}{\Delta(q)} = \frac{1}{q} + 744 + \cdots .$$

The series E_2 is not a modular form since (see [38] or Corollary 5.2.17):

Theorem 2.5. *For all matrices* $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ *in* Γ*, one has*

$$E_2((a\tau + b)/(c\tau + d)) = (c\tau + d)^2 E_2(\tau) + \frac{6c}{\pi i}(c\tau + d). \tag{5}$$

We can build a modular form easily as follows. Let N be an integer and let F_N denote the *multiplier* $E_2(\tau) - NE_2(N\tau)$. From [37] and [3], we get

Proposition 2.6. *The function* F_N *is a modular form of weight 2 and trivial multiplier system for* $\Gamma_0(N)$.

Proof: Write, for $ad - bc = 1$ and $N \mid c$, the value

$$E_2\left(N\frac{a\tau + b}{c\tau + d}\right) = E_2\left(\frac{a(N\tau) + Nb}{(c/N)(N\tau) + d}\right) = ((c/N)N\tau + d)^2 E_2(N\tau) - \frac{6c}{N\pi i}((c/N)N\tau + d)$$

which leads to

$$NE_2\left(N\frac{a\tau + b}{c\tau + d}\right) = N(c\tau + d)^2 E_2(N\tau) - \frac{6c}{\pi i}(c\tau + d).$$

Subtracting $E_2((a\tau + b)/(c\tau + d))$, we see that

$$F_N((a\tau + b)/(c\tau + d)) = (c\tau + d)^2 F_N(\tau). \square$$

Some identities are known for small values of N, for instance [30] for $N \in \{2, 4\}$; [3] for $N = 3$ and 11; [7, Thm 6.2], [4, Thm 3.7] for $N \in \{5, 7\}$. A very nice relation is [4, Thm 6.3]

$$F_7(q) = 6\left(\sum_{m,n=-\infty}^{\infty} q^{m^2+mn+2n^2}\right)^2.$$

2.3. **Formulas.** Reference is Proposition 2.4.1. When $f(q) = \sum_{n \geq n_0} a_n q^n$, we introduce the operator

$$f'(q) = \frac{1}{2i\pi} \frac{df}{d\tau} = q\frac{df}{dq} = \sum_{n \geq n_0} n a_n q^n. \tag{6}$$

Several identities are classical:

$$\Delta = \frac{E_4^3 - E_6^2}{1728}, \quad j = \frac{E_4^3}{\Delta}, \quad j - 1728 = \frac{E_6^2}{\Delta}, \tag{7}$$

$$\frac{j'}{j} = -\frac{E_6}{E_4}, \quad \frac{j'}{j-1728} = -\frac{E_4^2}{E_6}, \quad j' = -\frac{E_4^2 E_6}{\Delta}, \quad \frac{\Delta'}{\Delta} = E_2, \tag{8}$$

to which we add the Ramanujan differential system:

$$3{E_4}' = E_4E_2 - E_6, \quad 2{E_6}' = E_6E_2 - E_4^2, \quad 12{E_2}' = E_2^2 - E_4. \tag{9}$$

2.4. **Expressing modular forms as polynomials.** We extend a remark already done in [12] that uses the following result from [41, §5.6.2].

Theorem 2.7. *A modular form f of weight $2k$ with integer coefficients can be expressed as a polynomial with integer coefficients in E_4, Δ if k is even and E_4, Δ, E_6 otherwise. The number of coefficients in this polynomial is approximately $k/6$.*

We can apply this result to higher index Eisenstein series. For instance

$$E_8 = E_4^2, E_{10} = E_4E_6. \tag{10}$$

Lemma 2.8. *Let $k > 1$. Consider the equation*

$$2i_4 + 3i_6 + 6i_{12} = k \tag{11}$$

where i_4, i_{12} are positive integers and $i_6 \in \{0, 1\}$. Write $k = 2k_0 + \epsilon$ for $\epsilon \in \{0, 1\}$ and $m = k_0 - \epsilon$. All solutions (i_6, i_4, i_{12}) to equation (11) are

$$(\epsilon, m - 3j, j) \text{ for } 0 \leq j \leq j_{\max} := \lfloor m/3 \rfloor.$$

Proof: If k is even, we write $k = 2m$, which forces $i_6 = 0$. We rewrite (11) as $i_4 + 3i_{12} = m$ and the result follows. If $k = 2m + 1$, we need $i_6 = 1$ and (11) becomes $i_4 + 3i_{12} = m - 1$, which concludes the proof. □

Proof of the theorem: the expression we are looking for is

$$f = \sum_{i_4, i_6, i_{12}} c_{i_4, i_6, i_{12}} E_6^{i_6} E_4^{i_4} \Delta^{i_{12}}$$

for all positive indices satisfying $4i_4 + 6i_6 + 12i_{12} = 2k$ and $i_6 \in \{0, 1\}$. which is in fact equivalent to (11) and we apply the Lemma. □

For $w = 2k$, write

$$P_{w,j} = E_6^\epsilon E_4^{m-3j} \Delta^j = q^j + \sum_{n > j} g_{j,n} q^n$$

Algorithm 1: Evaluating all $P_{w,j}$'s.

Function *EvaluateAllPwj*$(E_4, E_6^\epsilon, \Delta, w, j_{\max})$
- **Input :** E_4, E_6, Δ; $w = 2k$, $j_{\max}$
- **Output:** $(P_{w,j})$ for $0 \le j \le j_{\max}$
- 0. compute m
- 1. [compute $P_{w,j} = E_6^\epsilon \cdot \Delta^j$ for all j]
- 1.1. $P_{w,0} \leftarrow E_6^\epsilon$
- 1.2. **for** $j \leftarrow 1$ **to** $j_{\max}$ **do**
  - $P_{w,j} \leftarrow \Delta \cdot P_{w,j-1}$
- 2. $Q[0] \leftarrow 1$; $Q[1] \leftarrow E_4$; $Q[2] \leftarrow E_4^2$; $Q[3] \leftarrow E_4 \cdot Q[2]$
- 3. $m' = m - 3j_{\max}$; $S \leftarrow Q[m']$
- 4. **for** $j \leftarrow j_{\max -1}$ **to** 0 **do**
  - $S \leftarrow Q[3] \cdot S$
  - // $S = E_4^{m-3j}$
  - $P_{w,j} \leftarrow P_{w,j} \cdot S$
- **return** $\{P_{w,j}\}$

where the coefficients $g_{j,n}$ are integers and do not depend on f. We may precompute all $P_{w,j}$'s for $0 \le j \le j_{\max} = \lfloor m/3 \rfloor$ using Algorithm 1. The cost of this algorithm is $3j_{\max}+2$ multiplications of series. We could improve on this using squarings in Step 1.2. If this is a one-time computation, Step 4 may not update and store the $P_{w,j}$'s.

The next two results are crucial for our forthcoming computations.

Proposition 2.9. *Let $f = \sum_{n\ge 0} f_n q^n$ be a modular form of weight $w = 2k$. With the notations of Lemma 2.8, there exist numbers c_j such that*

$$f = \sum_{j=0}^{j_{\max}} c_j P_{w,j}$$

where the c_j's are solution of a triangular linear system.

Proof: Write

$$\sum_{n\ge 0} f_n q^n = \sum_{j=0}^{j_{\max}} c_j \left(q^j + \sum_{n>j} g_{j,n} q^n \right),$$

the system giving the c_j's is triangular. Solving the system takes $O(j_{\max}^2) = O(w^2)$ operations with a very small constant. □

From this, we deduce:

Corollary 2.10. *When the f_n's are integers, so are the c_j's.*

Corollary 2.11. *To express f as a polynomial, we need all series developped at order $j_{\max} \approx w/12$.*

To prepare for the computation of modular equations of Section 5, we need to express a collection of modular forms of weight $2kr$ for $1 \leq r \leq \psi(N)$ (ℓ is an odd prime > 3) as polynomials in E_4, E_6 and Δ. The rationale is to share evaluations of products $P_{\epsilon,x,y} = E_6^\epsilon E_4^x \Delta^y$. It follows from Corollary 2.11 that the forms must be computed with order close to $w\psi(N)/12 = k\psi(N)/6$.

Using notations from Lemma 2.8 for $1 \leq r \leq \psi(N)$, write $kr = 2k_r + \epsilon_r$, $m_r = k_r - \epsilon_r$. Each m_r gives rise to a set of indices $(m_r - 3j_r, j_r)$ for $0 \leq j_r \leq m_r/3$. We can gather all these points in the plane to give rise to interesting patterns, as indicated in Figure 1. We define two sets of indices $\mathcal{I}_\epsilon$ corresponding to pairs $(m_r - 3j_r, j_r)$ with $\epsilon_r = \epsilon \in \{0, 1\}$. When k is even, $\mathcal{I}_1$ is empty.

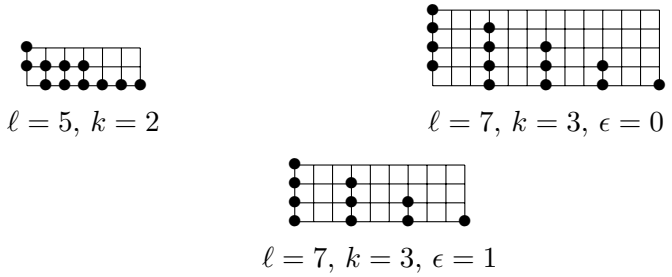

FIGURE 1. Examples of grids and sets $\mathcal{I}_\epsilon$.

Lemma 2.12. *The sets $\mathcal{I}_\epsilon$ satisfy the following properties.*

i) $\mathcal{I}_0$ is included in the grid $(0,0)\times(m_{\max,0}, j_{\max,0})$ with $m_{\max,0} = \lfloor k\psi(N)/2 \rfloor$, $j_{\max,0} = \lfloor k\psi(N)/6 \rfloor$;

ii) the distance between two consecutive abscissas in $\mathcal{I}_0$ is $k/2$ when k is even and k when k is odd;

iii) for any abscissa x of a point in $\mathcal{I}_0$, the maximal y is $\lfloor (m_{\max,0} - x)/3 \rfloor$. For k odd, $\mathcal{I}_1$ is such that

iv) $\mathcal{I}_1 \subset (0,0) \times (m_{\max,1}, j_{\max,1})$ with $m_{\max,1} = k(r' - 3)/2$ with r' the largest odd integer $\leq \psi(N)$ and $j_{\max,1} = \lfloor (k\ell - 3)/6 \rfloor$;

v) the distance between two consecutive abscissas in $\mathcal{I}_1$ is k;

vi) for any abscissa x in $\mathcal{I}_1$, the maximal y is $\lfloor (m_{\max,1} - x)/3 \rfloor$.

Proof: i) the maximal value of y is always reached for $m_{\max,0} = m_{\psi(N)} = \lfloor k\psi(N)/2 \rfloor$.

ii) When k is even, all ϵ_r's are 0, $m_r = k_r$, the minimal value is $m_1 = k/2$. Two consecutive values have distance $m_{r+1} - m_r = k_{r+1} - k_r = k/2$. When k is odd, the smallest m is $m_2 = k$. Two consecutive values are such that $m_{2r+2} - m_{2r} = k$.

iii) A point (x, y) is in $\mathcal{I}_0$ if and only if there exists $0 \leq m \leq m_{\max,0}$ and $0 \leq j \leq j_{\max,0}$ such that $(x, y) = (m - 3j, j)$. It follows that $j \leq (m - x)/3 \leq (m_{\max,0} - x)/3$.

iv) In the remainder of the proof, k is odd. When $\epsilon = 1$ (which cannot happen unless both k and r are odd) we get $m_r = (kr-3)/2$, and the maximal value is $m_{\max,1} = (kr'-3)/2$. Note that $r' = \ell$ when $N = \ell$ is prime.

v) The minimal value is $m_1 = \max(0, (k-3)/2)$. Two consecutive values are such that $m_{2r+3} - m_{2r+1} = k$.

vi) Proceed as in case iii. □

We write algorithms in a generic manner, and it will work for series, floating point numbers, etc. The auxiliary routine performing operations is given in Algorithm 2. We need one multiplication for each point in $\mathcal{I}_\epsilon$. Each power of E_4 is computed incrementally. The main function is given in Algorithm 3. The simple algorithm expressing the coefficients of the representation of f is give as Algorithm 4.

Algorithm 2: Compute all $P_{\epsilon,x,y}$'s.

Function *EvaluateAllPepsilon(ϵ, $E_4, E_6^\epsilon, \Delta$, $m_{\max}$, x_0, dx)*
- **Input :** E_4, E_6^ϵ and Δ, $m_{\max}$, x_0, dx
- **Output:** $(P_{\epsilon,x,y})$ for all points in $\mathcal{I}_\epsilon$
- 0. $j_{\max} \leftarrow \lfloor m_{\max}/3 \rfloor$
- 1. compute $P_{\epsilon,0,j}$ for all $0 \le j \le j_{\max}$
- 3. **for** $x \leftarrow x_0$ **to** $m_{\max}$ *by* dx **do**
  - compute $P_{\epsilon,x,y} = E_6^\epsilon E_4^x \Delta^y$ for all $0 \le y \le \lfloor (m_{\max} - x)/3 \rfloor$

Algorithm 3: Evaluating all $P_{\epsilon,x,y}$'s.

Function *EvaluateAllP(ψ, k, $E_4, E_6^\epsilon, \Delta$)*
- **Input :** E_4, E_6^ϵ and Δ, $\psi = \psi(N)$, k
- **Output:** $(P_{\epsilon,x,y})$ for all points in $\mathcal{I}_\epsilon$ and all ϵ
- 1. **if** k *is even* **then**
  - $dx \leftarrow k/2$
- **else**
  - $dx \leftarrow k$
- 2. $P_{0,x,y} \leftarrow$ EvaluateAllPepsilon(0, E_4, E_6, Δ, $k\psi/2$, dx, dx)
- 3. **if** k *is odd* **then**
  - $P_{1,x,y} \leftarrow$ EvaluateAllPepsilon(1, E_4, E_6, Δ, $k(r'-3)/2$, $(k-3)/2$, k) where r' is the largest odd integer $\le \psi$
- 4. **return** $\{P_{0,x,y}\} \cup \{P_{1,x,y}\}$.

2.5. **Modular polynomials from modular forms.** Let us begin with modular forms f for $\Gamma_0(N)$ of even weight w. In this section (R_i) denote a list of representatives of cosets of $\Gamma_0(N)\backslash\Gamma$.

Algorithm 4: Express f as a polynomial in the $P_{w,j}$'s.

Function *ExpressForm(f, $P_{w,j}$, $j_{\max}$)*
- **Input :** f a modular form of weight w
- **Output:** The coefficients of the representation of f as $\sum_{j=0}^{j_{\max}} c_j P_{w,j}$
- 0. $g \leftarrow f$
- 1. **for** $j \leftarrow 0$ **to** $j_{\max}$ **do**
  - // $g = g_j q^j + \cdots$
  - 1.1 $c_j \leftarrow g_j$
  - 1.2 $g \leftarrow g - c_j P_{w,j}$
- 2. **return** c_j's.

2.5.1. *General results.*

Theorem 2.13. *Let f be a modular form of weight w for $\Gamma_0(N)$. The values $f|_w(R_i)$ are conjugate over $\Gamma_0(N)$ and define a polynomial*

$$\Phi[f](X) = \prod_R (X - f|_w(R_i)) = X^{\psi(N)} + C_1(f)X^{\psi(N)-1} + \cdots + C_{\psi(N)}(f).$$

a) The coefficient $C_t(f)$ is a modular form of weight wt for Γ.
b) The polynomial $\Phi[f](X, E_4, E_6, \Delta)$ is homogeneous with weight $w(\ell+1)$.

Proof:
a) $C_t(f)$ is a symmetric function of the $f|_w(R_i)$'s. Having a matrix of Γ operate on the (R_i)'s leave them globally invariant, so that $C_t(f)$ is invariant under Γ.
b) is a consequence of a). □

In practice, it is customary to compute the power sums of roots of $\Phi[f]$:

$$S_t(f) = \sum_{i=1}^{\psi(N)} (f|_w(R_i))^t$$

for $1 \leq t \leq \psi(N)$. The sum $S_t(f)$ is also a modular form of weight wt for Γ, hence is expressible as a polynomial in (E_4, E_6, Δ). The functions $S_t(f)$ are more easily computed and once recognized as polynomials in (E_4, E_6, Δ), we can recover the coefficients of $\Phi[f]$ using Newton's formulas using $O(\mathsf{M}(\psi(N)))$ operations (in a field of characteristic 0 or larger than $\psi(N)$).

Let us turn towards a special case.

Proposition 2.14. *If f is a modular form of weight w for Γ, then the function $g(\tau) = f(N\tau)$ is a modular form of weight w for $\Gamma_0(N)$.*

Proof: let $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma_0(N)$, in other words $c = Nc'$ for some integer c'. Write

$$g(M\tau) = f(N(M\tau)) = f\left(\frac{(Na)\tau + (Nb)}{(Nc')\tau + d}\right).$$

Consider the matrix

$$U = \begin{pmatrix} a & bN \\ c' & d \end{pmatrix}$$

which belongs to Γ since $ad - b(Nc') = 1$. Using

$$M = U \cdot \begin{pmatrix} N & 0 \\ 0 & 1 \end{pmatrix},$$

we get

$$g(M\tau) = f(U(N\tau)) = (c'(N\tau) + d)^w f(N\tau) = (c\tau + d)^w g(\tau),$$

which proves invariance. □

Inspired by [15, Example 6.2.15].

Theorem 2.15. *Let (R_i) be a system of cosets for $\Gamma_0(N)\backslash\Gamma$, the $(g|_w R_i)$ are permuted by Γ and we let $\Phi[g](X)$ denote the modular polynomial*

$$\Phi[g](X) = \prod_i (X - g|_w R_i).$$

If $R = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ is a representative of a coset, the matrix $\begin{pmatrix} aN & bN \\ c & d \end{pmatrix}$ is equivalent to a matrix $\begin{pmatrix} A & B \\ 0 & D \end{pmatrix}$ with $A = \gcd(a, c)$, $AD = N$ and $0 \leq B < D$. Moreover

$$g|_w R(\tau) = D^{-w} f\left(\frac{A\tau + B}{D}\right).$$

Proof: If a coset is $R = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ with $d \mid N$, $d > 0$, c defined modulo N/d:

$$\begin{aligned} g|_w R(\tau) &= (c\tau + d)^{-w} g\left(\frac{a\tau + b}{c\tau + d}\right) \\ &= (c\tau + d)^{-w} f\left(\frac{aN\tau + bN}{c\tau + d}\right) \\ &= N^{-w/2} f|_w \begin{pmatrix} aN & bN \\ c & d \end{pmatrix}(\tau). \end{aligned}$$

The last matrix is equivalent to some $R'_c = \begin{pmatrix} A & B \\ 0 & D \end{pmatrix}$ with $AD = N$ and $0 \leq B < D$ given by Proposition 2.2, so that

$$\begin{aligned} g|_w R(\tau) &= N^{-w/2} f|_w \begin{pmatrix} A & B \\ 0 & D \end{pmatrix}(\tau) \\ &= N^{-w/2} \left(N^{w/2} D^{-w} f\left(\frac{A\tau + B}{D}\right)\right) \\ &= D^{-w} f\left(\frac{A\tau + B}{D}\right). \quad \square \end{aligned}$$

In order to handle series with integer coefficients, we scale all conjugates by multiplication by N^w.

2.5.2. *Orders and heights.* The traditional modular polynomial Φ_N^t has height $6\psi(N)\log N$ approximately [16, 43]. To get a general result for modular forms, we first estimate the order of the series needed.

For integer $t > 0$, write

$$f^t(q) = \sum_{n=0}^{\infty} \alpha(t,n) q^n.$$

Proposition 2.16. *Let $D \mid N$. Then*

$$S_{D,t}(f) = \sum_{B=0}^{D-1} f^t\left(\frac{A\tau + B}{D}\right) = D \sum_{n=0}^{\infty} \alpha(t, Dn) q^{An}.$$

Proof: using $z = q^{A/D}$:

$$S_{D,t}(f) = \sum_{B=0}^{D-1} \sum_{n=0}^{\infty} \alpha(t,n) z^n \zeta_D^{Bn} = \sum_{n=0}^{\infty} \alpha(t,n) z^n \sum_{B=0}^{D-1} \zeta_D^{Bn}.$$

If n is prime to D, the inner sum $\Sigma_D(n)$ is 0 using the properties of roots of unity. If $D \mid n$, the sum is D. Now, suppose that $g = \gcd(n, D)1$ with $1 < g <$, and $n' = n/g$ which is prime to D/g. Write $B = B' + xD'$, $0 \leq B' < D'$. We obtain

$$\Sigma_D(n) = \sum_{x=0}^{g-1} \sum_{B'=0}^{D'-1} \zeta_{D'}^{B'n' + xn'D'} = \sum_{x=0}^{g-1} \sum_{B'=0}^{D'-1} \zeta_{D'}^{B'} = 0.$$

Finally:

$$S_{D,t}(f) = D \sum_{n=0}^{\infty} \alpha(t, Dn) q^{An}. \quad \square$$

Proposition 2.17. *If f is a modular form of weight w for Γ. We need to develop f up to order $wN\psi(N)/12$ to compute $\Phi[f(N\tau)]$.*

Proof: we need to compute the expansion of the sum

$$S_t(f) = \sum_{D|N} (N/D)^{wt} S_{D,t}(f).$$

To get $\mathcal{O}$ terms in the sum, we need to develop the series up to order $N\mathcal{O}$. If we need $\mathcal{O}$ terms, we need to have $Nn \geq \mathcal{O}$.

Since $S_t(f)$ is a modular form of weight wt, we need $(wt)/12$ terms with a maximal $\mathcal{O} = w\psi(N)/12$, which yields the result. □

Now, we turn our attention to estimating the height of Φ. For this, we need general bounds on the coefficients of modular forms. This is [15, Theorem 9.2.1]:

Theorem 2.18. *Let $f(q) = \sum_{n\geq 0} a_n q^n$ be a modular form of weight w for Γ.*

a) if f is a cusp form, then $a(n) = O(n^{w/2})$.

b) if f is not cuspidal (i.e., $a_0 \neq 0$), then there are two positive constants C_1, C_2 s.t.

$$C_1 n^{w-1} \leq |a(n)| \leq C_2 n^{w-1}.$$

For instance, for Eisenstein series:

$$n^{w-1} \leq \delta_{w-1}(n) \leq \zeta(n-1) n^{w-1}.$$

We could use better estimates, see [15, Chapter 9].

We slightly generalize the computations in [12] to the case of an arbitrary form.

Proposition 2.19. *Let f be a modular form for Γ of weight w and $K > 0$ an integer, $C > 0$ two constants such that $|a(n)| \leq Cn^{K-1}$. The size of the sum $S_{\psi(N)}(f)$ is approximately $K\psi(N)\log(N\psi(N))$.*

Proof: Let us denote $\psi(N)$ by ψ for short. With $u = |q|$, we get

$$|f(q)| \leq \sum_{n=0}^{\infty} n^{K-1}u^n \leq C\sum_{n=0}^{\infty}(n+K-1)(n+K-2)\cdots(n+1)u^n = C(K-1)!(1-u)^{-K} := C_1(1-u)^{-K},$$

from which $|f^m(q)| \leq C_1^m(1-u)^{-Km}$ for all m. By Proposition 2.17, we need to compute f^ψ at order $d \approx wN\psi/12$. The largest coefficient is therefore of the order of

$$L = C_1^\psi \frac{(d+K\psi-1)!}{d!}.$$

Using Stirling's formula, we get

$$\log L \approx \psi \log C_1 + d\log\left(\frac{d+K\psi}{d}\right) + K\psi\log(d+K\psi).$$

Replacing d, we get

$$\log L \approx K\psi\log(N\psi) + C'\psi,$$

which yields the result. □

This is the largest quantity used for computing power sums, before going back to the coefficients of the polynomial $\Phi[f(N\tau)]$, whose sizes have the same order. We infer

Corollary 2.20. *The height of* $\Phi[f(N\tau)]$ *is approximately* $K\psi(N)\log(N\psi(N))$.

2.5.3. *The prime case.* In this section, we keep the notations f, w, etc.

Proposition 2.21. *When* $N=\ell$ *is prime, the roots of* $\Phi[f(\ell\tau)]$ *are*

$$f(\ell\tau) \text{ and } \ell^{-w}f((\tau+h)/\ell) \text{ for } 0\le h<\ell.$$

Proof: In case $N=\ell$, the system of cosets is given in Proposition 2.3.

When $c>0$, let $\overline{c}=1/c \bmod \ell$, so that $R'_c \sim_\Gamma \begin{pmatrix} 1 & \overline{c} \\ 0 & \ell \end{pmatrix}$ and

$$g|_w R'_c(\tau) = \ell^{-w} f\left(\frac{\tau+\overline{c}}{\ell}\right).$$

When $c=0$, $R'_0 = \begin{pmatrix} \ell & 0 \\ 0 & 1 \end{pmatrix}$ is already reduced and $g|_w R'_0(\tau) = f(\ell\tau)$.

Finally, $R'_\ell \sim_\Gamma \begin{pmatrix} 1 & 0 \\ 0 & \ell \end{pmatrix}$, from which

$$g|_w R'_\ell(\tau) = \ell^{-w} f\left(\frac{\tau}{\ell}\right). \quad \square$$

In practice, though, we prefer to scale all the conjugates by ℓ^w, so that we have to deal with integer coefficients in the expansions. By Corollary 2.20, the corresponding modular polynomial has height close to $2w(\ell+1)\log\ell$.

Proposition 2.22. *For* $t\ge 1$*, the power sum* $S_t(f)$ *has a* q*-expansion.*

Proof: with the notations of Proposition 2.16:

$$S_t(f) = \ell^{wt} f^t(\ell\tau) + S_{\ell,t}(f)$$

and both terms have a q-expansion. $\square$

2.5.4. *The case of* F_ℓ. Remember that $F_\ell(\tau) = E_2(\tau) - \ell E_2(\ell\tau)$ is a modular form of weight 2 for $\Gamma_0(\ell)$. We simplify the presentation using $F=F_\ell$ and replace $|_2$ by $|$. We need to compute all $F|(R\tau)$ for all R's from Proposition 2.3.

Proposition 2.23. *We have* $F|(R_0\tau) = F(\tau)$. *With the notations of Lemma 2.4, for* $0<c<\ell$*:*

$$F|(R_c\tau) = -\frac{1}{\ell}\, F\left(\frac{\tau+\overline{c}}{\ell}\right).$$

Also

$$F|(R_\ell\tau) = -\frac{1}{\ell}\, F\left(\frac{\tau}{\ell}\right).$$

Proof: Let us suppose that $c > 0$. Start with

$$E_2\left(\begin{pmatrix} 1 & 0 \\ c & 1 \end{pmatrix}\right)(\tau) = (c\tau+1)^2 E_2(\tau) + \frac{6c}{\pi i}(c\tau+1).$$

Plugging R'_c in (5), we find

$$E_2(\ell(R_c\tau)) = E_2(T_c(R'_c\tau)) = (c(R'_c\tau)+u)^2 E_2(R'_c\tau) + \frac{6c}{\pi i}(c(R'_c\tau)+u).$$

We simplify

$$c(R'_c\tau)+u = \frac{c\tau + c\overline{c} + \ell u}{\ell} = \frac{c\tau+1}{\ell}$$

which leads to

$$E_2(\ell(R_c\tau)) = \left(\frac{c\tau+1}{\ell}\right)^2 E_2\left(\frac{c\tau+1}{\ell}\right) + \frac{6c}{\pi i}\frac{c\tau+1}{\ell}.$$

We deduce that

$$F(R_c\tau) = E_2(R_c\tau) - \ell E_2(\ell(R_c\tau)) = (c\tau+1)^2\left(E_2(\tau) - \frac{1}{\ell}E_2(R'_c\tau)\right).$$

Remark that

$$F(R'_c\tau) = E_2(R'_c\tau) - \ell E_2(\tau+\overline{c}) = E_2(R'_c\tau) - \ell E_2(\tau)$$

so that

$$F(R_c\tau) = -\frac{1}{\ell}(c\tau+1)^2\ F\left(\frac{\tau+\overline{c}}{\ell}\right),$$

or

$$F|(R_c\tau) = -\frac{1}{\ell}\ F\left(\frac{\tau+\overline{c}}{\ell}\right).$$

The last case is that of R_ℓ, which needs

$$E_2\left(\begin{pmatrix} 0 & -1 \\ 1 & \ell \end{pmatrix}\right)(\tau) = (\tau+\ell)^2 E_2(\tau) + \frac{6}{\pi i}(\tau+\ell).$$

Performing computations the way we treated $c > 0$, we find

$$F|(R_\ell\tau) = -\frac{1}{\ell}\ F\left(\frac{\tau}{\ell}\right). \quad \square$$

Again, it is more convenient to scale all conjugates by $-\ell$, which gives us $-\ell F(\tau)$ and $F((\tau+h)/\ell)$ for all $0 \le h < \ell$. Remark that Proposition 2.22 applies too, *mutatis mutandis*.

3. Fast numerical evaluation of Eisenstein series

Since one of the methods for computing modular polynomials uses floating point evaluations, we give some algorithms to compute our functions.

3.1. **Jacobi θ functions.** The classical θ functions are:

$$\theta_2(q_1) = \sum_{n\in\mathbb{Z}} q_1^{(n+1/2)^2},\ \theta_3(q_1) = \sum_{n\in\mathbb{Z}} q_1^{n^2},\ \theta_4(q_1) = \sum_{n\in\mathbb{Z}} (-1)^n q_1^{n^2}.$$

Among many properties, one has

$$\theta_3^4(q_1) = \theta_4^4(q_1) + \theta_2^4(q_1).$$

The latter formula enables to concentrate on the evaluation of $\theta_{3,4}(q_1)$ as is done in [18].

Note also the following [18, Prop. 4]

Proposition 3.1.

$$\lim_{\mathrm{Im}(\tau)\to+\infty} \theta_3(\tau) = 1,\ \lim_{\mathrm{Im}(\tau)\to+\infty} \theta_4(\tau) = 1,\ \lim_{\mathrm{Im}(\tau)\to+\infty} \theta_2(\tau) = 0.$$

The quantities (see [23, §13.20])

$$a = \theta_2(q_1),\quad b = \theta_3(q_1),\quad c = \theta_4(q_1)$$

satisfy the following identities (among others)

$$E_4 = (a^8 + b^8 + c^8)/2,\ E_6 = (a+b)(b+c)(c-a)/2,\ \Delta = (abc/2)^8. \tag{12}$$

From which we deduce

$$\lim_{\mathrm{Im}(\tau)\to+\infty} E_4(\tau) = 1,\ \lim_{\mathrm{Im}(\tau)\to+\infty} E_6(\tau) = 1.$$

3.2. **Fast evaluation of $E_{2k}(q)$ for $k \geq 1$.** The θ functions that can be evaluated at precision N in time $O(\mathsf{M}(N)\sqrt{N})$ with q_1-expansions (see [22]) or faster in $O(\mathsf{M}(N)\log N)$ using [18] and also [31]. It follows that the quantities E_{2k} (for $k \geq 2$) can be evaluated at precision N in $O(\mathsf{M}(N)\log N)$ operations. As a consequence $j(q)$ can also be evaluated with the same complexity. This is also the case for η; in practice the lacunary properties of powers of η can also be used (see [42] and [15, Remark 2.1.27 with the indications therein]).

Evaluating E_2 is less obvious. However, hidden in the proof of [29, Thm 4] (thanks to [30] for highlighting this), we find

$$\frac{E_2E_4}{E_6} = \frac{{}_2F_1\left(\frac{13}{12}, \frac{5}{12}; 1; \frac{1728}{j}\right)}{{}_2F_1\left(\frac{1}{12}, \frac{5}{12}; 1; \frac{1728}{j}\right)} = 1 + \frac{720}{j} + \cdots$$

where the Gauss hypergeometric function is defined by

$${}_2F_1\left(a, b; c; x\right) = \sum_{k=0}^{\infty} \frac{(a)_k (b)_k}{(c)_k\, k!} x^k, |x| < 1$$

where $(a)_k = a(a+1)\cdots(a+k-1)$. By [45, 46, 32] and also [9], this function can be computed at precision N in $O(\mathsf{M}(N)(\log N)^2)$ operations. See also [28] for realistic computations. Other links with hypergeometric functions could be investigated (A. Bostan, personal communication).

Also, note that evaluating F_ℓ for small prime ℓ can be done using the special formulas we mentioned above.

3.3. **A multi-value approach.** In practice, a simpler approach yields the values E_{2k} of many k's with $k \geq 1$ in time $O(\mathsf{M}(N)\sqrt{N})$ based on [22]. The cost reduces to that of one series evaluation.

From [5], we take

$$(q;q)_\infty = \exp(-2i\pi\tau/24)\eta(q),$$

and for $k \geq 0$:

$$T_{2k}(q) = 1 + \sum_{n=1}^{\infty}(-1)^n \left\{(6n-1)^{2k} q^{n(3n-1)/2} + (6n+1)^{2k} q^{n(3n+1)/2}\right\}.$$

Note that $(q;q)_\infty = T_0(q)$.

Theorem 3.2 (Section 6, general formulas for $T_{2k}(q)/T_0(q)$ are also given)**.**

$$\frac{T_2(q)}{T_0(q)} = E_2,\ \frac{T_4(q)}{T_0(q)} = 3E_2^2 - 2E_4,\ \frac{T_6(q)}{T_0(q)} = 15E_2^3 - 30E_2E_4 + 16E_6.$$

If we need to compute E_2, E_4 and E_6, we see that it is enough to evaluate the series T_{2k} for $k \in \{0,1,2,3\}$ followed by a handful of multiplications and divisions as given in the preceding Theorem. Moreover, we can evaluate these series by sharing the common powers of q. These powers are evaluated at a reduced cost using [22, Algorithm2]. We give the modified procedure as algorithm 5. In Step 3.2.3, we have added the contribution $(6n \pm 1)^{2i}$ to each `T[i]`. We assume that the cost of multiplying by these small quantities is negligible. Were it not the case, we could use incremental computations of the polynomials $(6n \pm 1)^{2i}$. The cost of this algorithm reduces to that of one of the series, gaining a factor $kmax$.

Algorithm 5 uses the primitive in Algorithm 6. The reason of Step 4 is that $T[k]$ will be close to 1 when q is small, so that we may not want to add 1 right at the beginning and perhaps not in this function.

3.4. **The case of imaginary arguments.** In practice, it is easier to consider $\tau = \rho i$ for real $\rho \geq 1$. In that case, $1 > q_0 = \exp(-2\pi) = 0.001867\ldots \geq q = \exp(-2\pi\rho) > 0$. The functions E_2 and E_6 are increasing from $E_{2k}(q_0)$ to 1 (note that $E_6(q_0) = 0$ and $E_2(q_0) = 3/\pi$ from [19]); E_4 is decreasing from $E_4(q_0)$ to 1. This is important to note for the computations not to explode. Remember also that $j(i) = 1728$.

We turn to the precision needed for evaluating the functions T_{2k}. Let N denote an integer and $T_{2k,N}$ the truncated sum up to $n = N-1$. Since the series is alternating, we can bound the error using

$$\begin{aligned} |T_{2k}(q) - T_{2k,N}(q)| &\leq \{(6N-1)^{2k}\, q^{N(3N-1)/2} + (6N+1)^{2k}\, q^{N(3N+1)/2}\} \\ &\leq ((6N-1)^{2k} + (6N+1)^{2k})\, q^{N(3N-1)/2}. \end{aligned}$$

Since $0 < q < 1$, this gives us a very fast quadratic convergent series.

Algorithm 5: Combined evaluation of $T_{2k}(q)$.

Function *EvaluateManyT(q, N, kmax)*
Input : q, N, $kmax$
Output: $(T_{2k}(q))$ for $0 \le k \le kmax$
1. **for** $k := 0$ **to** $kmax$ **do**
   $T[k] \leftarrow 0$
2. $s \leftarrow 1$; $A \leftarrow \{1\}$; $Q[1] \leftarrow \{q\}$; $c \leftarrow 0$
3. **for** $n := 1$ *while* $n(3n+1)/2 \le N$ **do**
   $s \leftarrow -s$
   // $s = (-1)^n$
   3.1 $c \leftarrow c + 2n - 1$
   3.2 **for** $r := 1$ **to** 2 **do**
      3.2.1 **if** $r = 2$ **then**
         $c \leftarrow c + n$
         // $c = n(3n+1)/2$
      3.2.2 $q' \leftarrow$ FindPowerInTable(A, Q, c)
      3.2.3 $C \leftarrow (6n + (-1)^r)^2$
      3.2.4 **for** $k := 0$ **to** $kmax$ **do**
         $T[k] \leftarrow T[k] + sq'$
         if $k < kmax$ **then**
            $q' \leftarrow Cq'$
4. **for** $k := 0$ **to** $kmax$ **do**
   $T[k] \leftarrow T[k] + 1$
5. **return** T.

4. The polynomials of Fricke and Charlap/Coley/Robbins

4.1. **The work of Elkies.** An isogeny is associated with its kernel, or its polynomial description (called *kernel polynomial*). Given some finite subgroup F of $\mathcal{E}$, one can build an isogenous curve $\mathcal{E}^*$ and the corresponding isogeny, using Vélu's formulas. In the context of point counting, we discover a curve $\mathcal{E}^*$ that is ℓ-isogenous to $\mathcal{E}$ via its j-invariant as a root of the traditional modular polynomial, and we need to find the coefficients of $\mathcal{E}^*$, together with the isogeny. The idea of Elkies is to consider the same problems on the Tate curves associated to the elliptic curves $\mathcal{E}$ and $\mathcal{E}^*$.

To be brief, $\mathcal{E}$ has an equation in some parameter q, and the isogenous $\mathcal{E}^*$ is associated to parameter q^ℓ, where ℓ is the degree of the isogeny, which in our case is associated with a finite subgroup F of cardinality ℓ. To be more precise, we consider $\mathcal{E}$ has having equation $y^2 = x^3 + Ax + B$ with

$$A = -3E_4(q), B = -2E_6(q). \tag{13}$$

Algorithm 6: Finding c as a combination of known values.

Function *FindPowerInTable(A, Q, c)*
  Input : $A = \{a_1, \ldots, a_z\}$, Q such that for all i, $Q[a_i] = q^{a_i}$, c
  Output: q^c; A and Q are updated
  if $c = 1$ **then**
    $q' \leftarrow Q[1]$
  else if $c = 2a$ *with* $a \in A$ **then**
    $q' \leftarrow Q[a]^2$
  else if $c = a + b$ *with* $a, b \in A$ **then**
    $q' \leftarrow Q[a] \cdot Q[b]$;
  else if $c = 2a + b$ *with* $a, b \in A$ **then**
    $q' \leftarrow Q[a]^2 \cdot Q[b]$
  $A \leftarrow A \cup \{c\}$
  $Q[c] \leftarrow q'$
  return q'.

With a compatible scaling, we get the equation for $\mathcal{E}^* : y^2 = x^3 + A^*x + B^*$ with

$$A^* = -3\ell^4 E_4(q^\ell),\ B^* = -2\ell^6 E_6(q^\ell). \tag{14}$$

More importantly, writing κ_r for the power sums of the roots of the kernel polynomial, we have

$$\kappa_1 = \frac{\ell}{2}(\ell E_2(q^\ell) - E_2(q)) = -\frac{\ell}{2}F_\ell(q). \tag{15}$$

Beyond this, Elkies proved [20, formulas (66) to (69)]

Proposition 4.1.

$$A - A^* = 5(6\kappa_2 + 2A\kappa_0),$$
$$B - B^* = 7(10\kappa_3 + 6A\kappa_1 + 4B\kappa_0),$$

together with an induction relation satisfied by other κ_k *for* $k > 3$.

This can rephrased as (κ_1, A^*, B^*) is enough to describe an isogeny. Also A^* and B^* belong to $\mathbb{Q}[\kappa_1, A, B]$ since κ_2 and κ_3 do. The minimal polynomial of $2\kappa_1$ is the modular polynomial associated to F_ℓ, and we can express A^* and B^* as elements in the field $\mathbb{Q}[\kappa_1, A, B]$, which we use below. Rephrased another times, $E_4(q^\ell)$ and $E_6(q^\ell)$ are modular forms we need to express as expressions in known modular forms. See [20] for more details on this subject.

Given these quantities, there are several algorithms to get the isogeny. We refer to [8] for this.

4.2. **The Fricke polynomials.**

4.2.1. *Reinterpreting Elkies's results.* One way of looking at the work of Elkies (taken from [12], but which originated in [26]) is to realize that we try to decompose the ℓ-th division polynomial f_ℓ (say ℓ is odd) of degree $(\ell^2-1)/2$ over a subfield of degree $\ell+1$. In Fricke's term, we compute a degree $\ell+1$ resolvent for the equation $f_\ell(X)=0$.

$$\begin{array}{c} \mathbb{Q}(A,B)[X]/(f_\ell(X,A,B)) \\ \Big| \ (\ell-1)/2 \\ \mathbb{Q}(A,B)[X]/(U_\ell(X,A,B)) \\ \Big| \ \ell+1 \\ \mathbb{Q}(A,B) \end{array}$$

4.2.2. *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 $\mathcal{E}^*: y^2=x^3+A^*x+B^*$. The results for U_ℓ are due to Fricke, and Charlap/Coley/Robbins for V_ℓ and W_ℓ.

Theorem 4.2. *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 $U_\ell(2\kappa_1,A,B)=0$, respectively $V_\ell(A^*,A,B)=0$, $W_\ell(B^*,A,B)=0$.*

Let us turn our attention to the properties of these polynomials. Note that U_ℓ (resp. V_ℓ and W_ℓ) is the minimal polynomial of a weight 2 form (resp. 4 and 6). We note ϖ the corresponding weight.

Theorem 4.3. *When $\ell>3$, the polynomials U_ℓ, V_ℓ, W_ℓ live in $\mathbb{Z}[X,Y,Z]$.*

As a consequence of Theorem 2.13, we have

Proposition 4.4. *The polynomials U_ℓ, V_ℓ and W_ℓ are homogeneous with weight $\varpi(\ell+1)$.*

Proposition 4.5. *Put $z^\ell=q=\exp(2i\pi\tau)$ and ζ_ℓ a root of unity. Then*

1) The roots of $U_\ell(X,A(q),B(q))$ are $F_\ell(z\zeta_\ell^h)$ for $0\le h<\ell$, and $-\ell F_\ell(q)$.

2) The roots of $V_\ell(X,E_4(q),E_6(q))$ (resp. W_ℓ) are $E_4(z\zeta_\ell^h)$ (resp. $E_6(z\zeta_\ell^h)$) for $0\le h<\ell$, and $\ell^4E_4(q^\ell)$ (resp. $\ell^6E_6(q^\ell)$).

Part 1 is proven in Proposition 2.23; part 2 is done in Proposition 2.21. We can also use Proposition 2.19 to get

Proposition 4.6. *The height of U_ℓ (resp. V_ℓ, W_ℓ) is approximately $\varpi(\ell+1)\log\ell$.*

4.2.3. *Representing A^* and B^* as rational fractions.* Once we have computed U_ℓ, we can either compute V_ℓ and W_ℓ or use another representation. From [35, Theorem 3.9], there exist polynomials $\mathcal{A}_\ell$ and $\mathcal{B}_\ell$ of degree less

than $\ell+1$ such that

$$A^* = \frac{\mathcal{A}_\ell(X,A,B)}{U'_\ell(X)},\ B^* = \frac{\mathcal{B}_\ell(X,A,B)}{U'_\ell(X)} \tag{16}$$

(Only here: $U'_\ell(X) = \frac{\partial U_\ell}{\partial X}$.) Moreover, $\mathcal{A}_\ell$ and $\mathcal{B}_\ell$ are polynomials with integer coefficients and of respected generalized weight $2\ell+4$ and $2\ell+6$. These formulas are of independent interest and may prove useful in other contexts. The authors of the reference use Groebner basis computations to find the two numerators. We propose another route later on.

4.2.4. *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 a single one in the traditional case. 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 $(2\kappa_1, z_1, z_2)$ we need to test whether this leads to an isogeny or not. See techniques for this task in [8]. Using the rational fractions for A^* and B^* is faster, just needing evaluations of rational fractions. A method based on using U_ℓ only is described in Section 6.

5. Computing Fricke polynomials

The authors of [12] give two methods of computation using manipulations of q-expansions of series over $\mathbb{Q}$. Following Atkin [1], these can be replaced by computations modulo small primes (preferably with convenient FFT multiplication) followed by recovery using the Chinese remaindering theorem using the bounds in Proposition 4.6. To this, we add two evaluation-interpolation algorithms already used for classical modular polynomials: the first is based on floating point calculations, the second on isogeny volcanoes.

The polynomials U_ℓ, V_ℓ and W_ℓ are modular polynomials of modular forms, so that they obey Theorem 2.13. Also

$$\mathcal{A}_\ell(X,Y,Z) = \sum_{r=0}^{\ell} X^r \sum_{2i_2+3i_3=\ell+2-r} a_{r,i_2,i_3} Y^{i_2} Z^{i_3},$$

$$\mathcal{B}_\ell(X,Y,Z) = \sum_{r=0}^{\ell} X^r \sum_{2i_2+3i_3=\ell+3-r} b_{r,i_2,i_3} Y^{i_2} Z^{i_3}.$$

All the methods to be described can be applied to U_ℓ, V_ℓ, W_ℓ, and also to $\mathcal{A}_\ell$, $\mathcal{B}_\ell$. To simplify the presentation, we assume from now on (unless indicated) that $\ell > 3$ and concentrate on U_ℓ, indicating what has to be changed for the other polynomials; in particular we assume we are looking for the modular polynomial of a form of weight w.

We rewrite

$$U_\ell(X) = X^{\ell+1} + C_1(E_4,E_6,\Delta)X^\ell + \cdots + C_{\ell+1}(E_4,E_6,\Delta).$$

By Theorem 2.13, C_t is a modular form for Γ of weight $wt = 2t$; this implies $C_1 = 0$. By (10), $C_2 = c_2 E_4$, $C_3 = c_3 E_6$, $C_4 = c_4 E_4^2$, $C_5 = c_5 E_4 E_6$, $C_6 = c_{6,1} E_4^3 + c_{6,0}\Delta$. For instance, the following methods will give us

$$U_5(X) = X^6 - 60E_4X^4 - 320E_6X^3 - 720E_4^2X^2 - 768E_4E_6X - 320E_4^3 + 552960\Delta.$$

5.1. **Using q-expansions.** Note that

$$2\kappa_1(q) = -\ell F_\ell(q) = \ell(\ell-1) + 24\ell \sum_{n=1}^{\infty} \delta_1'(n) q^n$$

where $\delta_1'(n)$ is the sum of the divisors of n prime to ℓ.

Using Proposition 4.5, we denote by $\sigma_t(q)$ the corresponding power sums of roots of U_ℓ:

$$\sigma_t(q) = (-\ell F_\ell(q))^t + \sum_{h=0}^{\ell-1} F_\ell(z\zeta_\ell^h)^t$$

for $z^\ell = q$ and $1 \le t \le \ell + 1$. We compute them and recover the coefficients of U_ℓ using Newton's formulas as explained in Section 2.5.

The power sums $\sigma_t(q)$ are modular forms of weight wt and can be represented as polynomials in E_4, E_6, Δ

$$\sigma_t(q) = \sum_j c_{t,j} P_{wt,j}$$

using Proposition 2.9 and Algorithm 3. This leads to a triangular linear system $\mathcal{S}_t$ in the $c_{t,j}$'s. The system has $\approx (wt/6)$ rows and can be solved with $O((wt)^2)$ operations over $\mathbb{Z}$, for a total of $O(\sum_t (wt)^2) = O(\ell^3)$. Once solved for all t's, we use Newton's identities to recover the coefficients of U_ℓ.

To start the process, one needs to evaluate the series $\sigma_t(q)$ using intermediate expressions in $z = q^{1/\ell}$ having roots of unity ζ_ℓ temporarily appearing and vanishing. See [21, §2.2] for more details and complexity analysis. In particular, if we denote by $\mathsf{M}_q(d)$ the number of arithmetic operations in $\mathbb{Z}$ required to multiply two dense q-expansions with d terms, then the total complexity of the series computations is $O(\ell \mathsf{M}_q(\ell d))$, which is $O(\ell \mathsf{M}_q(\ell^2))$ in our case. If H is a bound on the height of the polynomial, then the bit complexity is $O(\ell^3 (\log \ell) \mathsf{M}(H))$. Assuming $H \in O(\ell \log \ell)$ by Proposition 4.6, this is $O(\ell^4 \log^{3+\epsilon} \ell)$.

Example. Consider the case $\ell = 5$. The systems $\mathcal{S}_t$ to be solved come from the equations:

$$\begin{aligned}
\sigma_2(q) &= c_{2,0} E_4, \\
\sigma_3(q) &= c_{3,0} E_6, \\
\sigma_4(q) &= c_{4,0} E_4^2, \\
\sigma_5(q) &= c_{5,0} E_6 E_4, \\
\sigma_6(q) &= c_{6,0} E_4^3 + c_{6,1}\Delta.
\end{aligned}$$

We compute

$$\sigma_6(q) = 1000320 + 186071040q + \cdots$$

and we remember that $E_4(q) = 1 + 240q + \cdots$, $\Delta(q) = q + \cdots$ so that the system $\mathcal{S}_6$ is

$$\begin{cases} 1000320 &= c_{6,0} \\ 186071040 &= 720\, c_{6,0} + c_{6,1} \end{cases}$$

which is triangular indeed and therefore easy to solve. Its solutions are integers.

We can also work modulo small primes and use the Chinese Remainder Theorem to recover the polynomials.

5.2. **Floating point methods.** We adapt the methods proposed for ordinary modular equations to our polynomials (U_ℓ, V_ℓ, W_ℓ). We note H for the logarithmic height of the polynomials, that we have estimated to $\varpi(\ell+1)\log\ell$ in Proposition 4.6. All the methods are heuristic.

5.2.1. *Solving a linear system.* We start from $U_\ell(2\kappa_1(q), E_4(q), E_6(q), \Delta(q)) = 0$ and we compute floating point values to get a linear system in the coefficients that should come out as integers for $\ell > 3$. We evaluate $\kappa_1(q)$, $E_4(q)$ and $E_6(q)$ (and therefore $\Delta(q)$ at high precision for chosen values of (imaginary) τ in $q = \exp(2i\pi\tau)$. This would involve $O(\ell^{2\omega})$ floating point operations, and we can do better in the following section.

5.2.2. *Using power sums.* The case of the traditional modular polynomial is treated in [21]. We can use the same approach for our polynomials. First of all, we need to compute

$$-\ell F_\ell(q), \quad \{F_\ell(z\zeta_\ell^h), 0 \le h < \ell\},$$

where $z^\ell = q$ and ζ_ℓ is a primitive ℓ-th root of unity. By definition

$$F_\ell(q) = E_2(q) - \ell E_2(q^\ell)$$

and

$$F_\ell(z\zeta_\ell^h) = E_2(z\zeta_\ell^h) - \ell E_2(q),$$

and the last term is a constant w.r.t. h. We first evaluate $E_2(q)$, $E_2(q^\ell)$ and then the other roots, by sharing the computations: All terms we need are of the form $(z\zeta_\ell^h)^e = z^e\zeta_\ell^{(he) \bmod \ell}$. When $\ell \mid e$, the computation is a little faster. We give the corresponding code as Algorithm 7. We also precompute $\xi_h = \zeta_\ell^h$. The complete code is in Algorithm 8. Multiple evaluations of $\eta(k\tau)$ can be shared as explained in [22].

The system $\mathcal{S}_t$ has size $O(t^2)$ and we need $O(t^\omega)$ operations to solve it, for a total of $O(\ell^{\omega+1})$. Like in the series case, anticipate integer coefficients, which makes recognition of the coefficients easier.

Example. Take again $\ell = 5$, for which the $\mathcal{S}_t$ were already given. Let us concentrate on the case of $\sigma_6(q) = u_{6,0}E_4^3 + u_{6,1}\Delta$; we start with $\mathcal{L}_6 = \emptyset$. Using $\rho = 1.1$ leads to

$$\mathcal{L}_6 = \{1.912407642u_{6,0} + 0.0009726854527956u_{6,1} = 1393450.57337539139\}$$

Algorithm 7: Combined evaluation of $T_{2k}(z\zeta_\ell^h)$.

Function *EvaluateConjugateValues(ℓ, z, N, (ξ), $kmax$)*
- **Input :** ℓ, z, (ξ), N
- **Output:** $(T_{2k}(z\zeta_\ell^h))$ for $0 \le k \le kmax$, $0 \le h < \ell$
- 1. **for** $k := 0$ **to** $kmax$ **do**
  - **for** $h := 0$ **to** $\ell - 1$ **do**
    - $T[k,h] \leftarrow 0$
- 2. $s \leftarrow 1$; $A \leftarrow \{1\}$; $Z[1] \leftarrow \{z\}$; $c \leftarrow 0$
- 3. **for** $n := 1$ *while* $n(3n+1)/2 \le N$ **do**
  - $s \leftarrow -s$
  - // $s = (-1)^n$
  - 3.1 $c \leftarrow c + 2n - 1$
  - 3.2 **for** $r := 1$ **to** 2 **do**
    - 3.2.1 **if** $r = 2$ **then**
      - $c \leftarrow c + n$
      - // $c = n(3n+1)/2$
    - 3.2.2 $z' \leftarrow s\cdot$FindPowerInTable(A, Z, c)
    - 3.2.3 $C \leftarrow (6n + (-1)^r)^2$
    - 3.2.4 **for** $k := 0$ **to** $kmax$ **do**
      - $T[k,0] \leftarrow T[k,0] + z'$
      - **for** $h := 1$ **to** $\ell - 1$ **do**
        - $T[k,h] \leftarrow T[k,h] + \xi[(hc) \bmod \ell]\, z'$
      - **if** $k < kmax$ **then**
        - $z' \leftarrow C \cdot z'$
- 4. **for** $k := 0$ **to** $kmax$ **do**
  - **for** $h := 0$ **to** $\ell - 1$ **do**
    - $T[k,h] \leftarrow T[k,h] + 1$
- 5. **return** T.

and the following iteration with $\rho = 1.2$ adds

$$\{1.435895343u_{6,0} + 0.0005247501300701u_{6,1} = 1156054.63606077432\}$$

and the solution of $\mathcal{L}_6$ (rounded to integers) is

$$u_{6,1} = -534159360, u_{6,0} = 1000320.$$

5.3. **Isogeny volcanoes.** The method in [13] shares many common points with the method to be described next but with a worse complexity. It uses supersingular curves whose complete explicit ℓ-torsion is required. The work of [10] is a building block in [43] where direct evaluation of $\Phi_\ell(X, j(E))$ mod q is made possible using an explicit version of the Chinese remainder theorem

Algorithm 8: Computing Fricke polynomial using floating point numbers.

Function *ComputeUVW(ℓ, w)*
- **Input :** $w \in \{2, 4, 6\}$ corresponding to U_ℓ, V_ℓ or W_ℓ respectively, ℓ an odd prime
- **Output:** the corresponding Fricke polynomial
- 0.0 $H \leftarrow w(\ell+1)\log \ell$; all computations are carried out at precision H
- 0.1 compute $\zeta_\ell \leftarrow \exp(2i\pi/\ell)$
- 0.2 **for** $h := 0$ **to** $\ell - 1$ **do**
  - $\xi_h \leftarrow \zeta_\ell^h$
- 0.3 Compute all systems $\mathcal{S}_t$ for $2 \le t \le \ell + 1$
- 0.4 **for** $t := 2$ **to** $\ell + 1$ **do**
  - $\mathcal{L}_t \leftarrow \emptyset$
- 0.5 $\rho \leftarrow 1$
- 1. **while** *there is a system $\mathcal{L}_t$ that is not solved* **do**
  - 1.0 $\rho \leftarrow \rho + 0.1$
  - 1.1 $z \leftarrow \exp(-2\pi\rho/\ell)$; $q_\rho \leftarrow z^\ell$
  - 1.2 $T \leftarrow$ EvaluateConjugateValues(ℓ, z, N, (ξ_h), s)
  - 1.3. use Theorem 3.2 to evaluate E_{2k} for all q_i's from T, yielding $(\kappa(q_i))$ for $i = 0, \ldots, \ell + 1$; also deduce $E_{4,\rho} = E_4(q_\rho)$, $E_{6,\rho} = E_6(q_\rho)$, $\Delta_\rho = (E_{4,\rho}^3 - E_{6,\rho}^2)/1728$
  - 1.4 **for** $r := t$ **to** $\ell + 1$ **do**
    - **if** $\mathcal{L}_t$ *is not solved* **then**
      - 1.4.1 Instantiate $\mathcal{S}_t$ with $\sum_i \kappa_1(q_i)^t$, $E_{4,\rho}$, $E_{6,\rho}$, Δ_ρ; add it to $\mathcal{L}_t$
      - 1.4.2 **if** $\mathcal{L}_t$ *has as many equations as unknowns* **then**
        - solve $\mathcal{L}_t$ and store the values; declare $\mathcal{L}_t$ solved
- 2. Round the coefficients and use Newton's formulas.

modulo small primes. Our version is an adaptation to the computation of the Fricke polynomials.

5.3.1. *Quick presentation.* In a nutshell, the algorithm in [10] performs computations modulo special primes p satisfying arithmetical conditions: $p \equiv 1 \bmod \ell$ and $4p = t^2 - \ell^2 v^2 D$ in integers t and v, D not a multiple of ℓ; $D < 0$ is the (fundamental) discriminant of some auxiliary quadratic field. With these conditions, the so-called class polynomial $H_D(z)$ splits completely modulo p and its roots are j-invariants of elliptic curves with complex multiplication by the maximal order $\mathcal{O}_D$. The isogeny volcanoes that we can build have only one level (see Figure 2) and the corresponding j-invariants are the roots of $H_{\ell^2 D}(X)$. Basically, the algorithm interpolates

data using the isogenies attached to the volcanoes. We refer the reader to the original article for more properties related to elliptic curves. For our purpose, we just need to know that we have isogeny data available and that they can help us computing the polynomial $U_\ell(X, E_4, E_6, \Delta) \bmod p$ from these data. We refer to the article for the complexity under GRH, namely $O(\ell^3(\log \ell)^3 \log\log \ell)$ using $O(\ell^2 \log(\ell p))$ space for suitably chosen p.

FIGURE 2. A typical set of volcanoes.

We adapt a slight modification of the simplified version Algorithm 2.1 of [10] to our needs to give Algorithm 9. All we describe is also valid in the full version in [10].

5.3.2. *The algorithm for U_ℓ.* We denote by $\mathcal{P}_r$ the power sums of U_ℓ. Such a $\mathcal{P}_r$ is a modular form of weight wr for Γ. As explained in Section 2.4, we may write these power sums as

$$\mathcal{P}_r(E_4, E_6, \Delta) = \sum_{j_r=0}^{m_r} c_{r,j_r} P_{wr,j}.$$

The values of $P_{wr,j}$ and the sums will be reconstructed from values $\kappa_{1,i}$ associated to curves $\mathcal{E}_i : y^2 = x^3 + A_i x + B_i$.

Note that for each i, the kernel polynomial of the isogeny from $\mathcal{E}_i$ to $\mathcal{E}'_{ik}$ starts $X^{(\ell-1)/2} - \kappa_{1,i} X^{(\ell-3)/2} + \cdots$, so that

$$U_\ell(X, E_{4,i}, E_{6,i}, \Delta_i) = \prod_{i=1}^{\ell+1} (X - \kappa_{1,i}).$$

Given these roots, it is easy to compute the power sums, see Algorithm 10. Trading multiplications for additions, Step 2 costs $O(\ell^2)$ operations over $\mathbb{F}_p$.

For V_ℓ (resp. W_ℓ), replace κ_1 by A^* (resp. B^*) in Step 4 as far as reconstruction is concerned.

A numerical example: Let us give one value for $\ell = 5$. We select $D = -71$ for which $h(-71) = 7 \geq 5 + 2$. Consider $p = 1811$. The roots of $H_{-71}(z)$ modulo p are:

$$\mathcal{J}_D = \{313, 1073, 1288, 1312, 1402, 1767, 1808\}.$$

Algorithm 9: The core algorithm.

Function *PartialVolcano(ℓ, D, $H_D(X)$, p)*

Input : ℓ an odd prime, D the discriminant of an imaginary quadratic order $\mathcal{O}$ with class number $h(D) \geq \ell + 2$; H_D the class polynomial associated to the order $\mathcal{O}$; p prime with $p \equiv 1 \bmod \ell$ and $4p = t^2 - \ell^2 v^2 D$, $v \not\equiv 0 \bmod \ell$

Output: A collection $(\mathcal{E}_i, \{Q_{ik}, \mathcal{E}'_{ik}\}_{1 \leq k \leq \ell+1})_{1 \leq i \leq h}$ where $\mathcal{E}'_{ik} = \mathcal{E}_i / \langle Q_{ik} \rangle$

1. Build the list $\mathcal{J}_D$ containing the roots of $H_D(z)$ modulo p
2. **for** $j_i \in \mathcal{J}_D$ **do**
   - 2.1 find a curve $\mathcal{E}_i : y^2 = x^3 + A_i x + B_i$ having invariant j_i and cardinality $m = p + 1 - t$;
   - 2.2 find all the neighbors $\mathcal{N}(\mathcal{E}_i)$ in the volcano of $\mathcal{E}_i$: 2 horizontal isogenies and $\ell - 1$ on the floor. Let $\mathcal{K}$ be a set to contain invariants and initialized to $\emptyset$.
   - **while** *we do not have all isogenies of both kinds* **do**
     - 2.2.1 Select a random point Q_{ik} of order ℓ on $\mathcal{E}_i / \mathbb{F}_p = [A_i, B_i]$.
     - 2.2.2 Compute the rational isogeny $\mathcal{E}_i \to \mathcal{E}'_{ik} = \mathcal{E}_i / \langle Q_{ik} \rangle$ using Vélu's formulas. The result is a pair $(Q_{ik}, \mathcal{E}'_{ik})$.
     - 2.2.3 **if** $j(\mathcal{E}'_{ik}) \notin \mathcal{K}$ **then**
       - $\mathcal{K} \leftarrow \mathcal{K} \cup \{j(\mathcal{E}'_{ik})\}$
       - if $j(\mathcal{E}'_{ik})$ is a root of H_D, then $\mathcal{E}'_{ik}$ is on the crater and is one of the two neighbours. If it does not belong to the crater, it belongs to the floor. Store the pair $(Q_{ik}, \mathcal{E}'_{ik})$.
   - Store $\mathcal{E}_i, \{Q_{ik}, \mathcal{E}'_{ik}\}_{1 \leq k \leq \ell+1}$
3. **return** $(\mathcal{E}_i, \{Q_{ik}, \mathcal{E}'_{ik}\}_{1 \leq k \leq \ell+1})_{1 \leq i \leq h}$.

Associated are curves and neighbors for each j value. These can be found in Table 1. The power sums $\mathcal{P}_r$ corresponding to the values are:

$\mathcal{E}_i \backslash r$	1	2	3	4	5	6
$[1582, 902]$	0	105	1680	1379	756	772
$[1662, 405]$	0	527	1188	90	748	888
$[1451, 1331]$	0	1723	403	350	293	583
$[1013, 747]$	0	1133	18	1594	1738	105
$[224, 753]$	0	95	760	1790	1603	27
$[1128, 1504]$	0	155	669	1424	1130	522
$[91, 725]$	0	1793	1523	589	1233	134

Algorithm 10: Computing $U_\ell(X,Y,Z) \bmod p$.

Function ComputeUMod(ℓ, D, $H_D(z)$, p):

Input : ℓ an odd prime, D the discriminant of an imaginary quadratic order $\mathcal{O}$ of discriminant D with class number $h(D) \geq \ell+2$; H_D is the class polynomial associated to order $\mathcal{O}$; p prime with $p \equiv 1 \bmod \ell$ and $4p = t^2 - \ell^2 v^2 D$, $v \not\equiv 0 \bmod \ell$

Output: $U_\ell(X,Y,Z) \bmod p$

1. $(\mathcal{E}_i, \{Q_{ik}, \mathcal{E}'_{ik}\}_{1\leq k\leq \ell+1})_{1\leq i\leq h} \leftarrow$PartialVolcano($\ell$, D, $H_D(X)$, p)
2. Evaluate the quantities $P_{wt,j}$ in the $\{\mathcal{E}'_{ik}\}$ using Algorithm 3
3. **for** $i \leftarrow 1$ **to** $\ell+1$ **do**
   for $t \leftarrow 1$ **to** $\ell+1$ **do**
   $\sigma_{t,i} \leftarrow (1/2)\sum_{k=1}^{\ell-1} x(Q_{ik})^t$
4. **for** $t \leftarrow 1$ **to** $\ell+1$ **do**
   solve the linear system
   $$\sigma_{t,i} = \sum_{j=0}^{m_t} c_{t,j} P_{t,j}.$$
5. **return** U_ℓ *recovered from* c_{t,j_t}*'s using Newton's formulas.*

For instance, $\sigma_6 = c_{6,0}E_4^3 + c_{6,1}\Delta$, we need to solve

$$\begin{cases} 772 &=& c_{6,0}680^3 + c_{6,1}1067 \bmod 1811, \\ 888 &=& c_{6,0}1257^3 + c_{6,1}874 \bmod 1811, \\ 583 &=& c_{6,0}120^3 + c_{6,1}363 \bmod 1811, \\ \cdots & \cdots & \cdots \end{cases}$$

that is $648E_4^3 + 523\Delta$. The coefficients are:

$$\begin{aligned} \sigma_2 &= 120E_4 \\ \sigma_3 &= 960E_6 \\ \sigma_4 &= 1025E_4^2 \\ \sigma_5 &= 235E_6E_4 \\ \sigma_6 &= 648E_4^3 + 523\Delta \end{aligned}$$

5.3.3. *Computing $\mathcal{A}_\ell$ and $\mathcal{B}_\ell$.* In this section, we use A and B instead of (E_4, E_6, Δ) for ease of presentation. Once U_ℓ is available, we can use equation (16) in which we plug the series to get

$$-3\ell^4 E_4(q^\ell) \cdot U'_\ell(\sigma_1(q), A(q), B(q)) = \mathcal{A}_\ell(\sigma_1(q), A(q), B(q)). \tag{17}$$

Similarly, we would use

$$-2\ell^6 E_6(q^\ell) \cdot U'_\ell(\sigma_1(q), A(q), B(q)) = \mathcal{B}_\ell(\sigma_1(q), A(q), B(q))$$

to compute $\mathcal{B}_\ell$.

We find the coefficients by solving a linear system (over $\mathbb{Q}$ or using small primes as already described). We can precompute the powers of the series for σ_1, A and B and remark that U_ℓ and $\mathcal{A}_\ell$ share a lot of them. Also, the series $E_4(q^\ell)$ is rather sparse, so that the product with this quantity is fast. There is an advantage to compute $\mathcal{A}_\ell$ and $\mathcal{B}_\ell$ at the same time, sharing as many powers as possible.

Let us turn our attention towards the computations of $\mathcal{A}_\ell$ (resp. $\mathcal{B}_\ell$) using evaluation/interpolation methods. There is nothing special about using floating point numbers, except that the system we have to solve has size $O(\ell^2 \times \ell^2)$ leading to a $O(\ell^{2\omega})$ time algorithm.

Some care must be taken when using the isogeny approach. To exemplify the problem, consider the case $\ell = 11$ (similar problems do not occur for smaller ℓ's). The polynomial $\mathcal{A}_{11}$ reads:

$$\mathcal{A}_{11} = a_{1,1,0}(X^{11}A)+ + \cdots + X(a_{11,6,0}A^6 + a_{11,3,2}A^3B^2 + a_{11,0,4}B^4) + (a_{12,5,1}A^5B + a_{12,2,3}A^2B^3).$$

To find them, we use a 20×20 system whose rightmost columns are

$$M = \begin{pmatrix} \cdots & \kappa_{1,1}A_1^6 & \kappa_{1,1}A_1^3B_1^2 & \kappa_{1,1}B_1^4 & A_1^5B_1 & A_1^2B_1^3 \\ & & \cdots & & & \\ \cdots & \kappa_{1,12}A_1^6 & \kappa_{1,12}A_1^3B_1^2 & \kappa_{1,12}B_1^4 & A_1^5B_1 & A_1^2B_1^3 \\ \cdots & \kappa_{1,13}A_2^6 & \kappa_{1,13}A_2^3B_2^2 & \kappa_{1,13}B_2^4 & A_2^5B_2 & A_2^2B_2^3 \\ & & \cdots & & & \\ \cdots & \kappa_{1,20}A_2^6 & \kappa_{1,20}A_2^3B_2^2 & \kappa_{1,20}B_2^4 & A_2^5B_2 & A_2^2B_2^3 \end{pmatrix}$$

The first 12 rows ($12 = \ell + 1$) are the $\ell + 1$ curves isogenous to $[A_1, B_1]$. The remaining 8 are taken from the $\ell + 1$ curves isogenous to $[A_2, B_2]$. Consider the product

$$M \times \begin{pmatrix} 0 \\ 0 \\ \cdots \\ 0 \\ 1 \\ u \\ v \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \kappa_{1,1}(A_1^6 + uA_1^3B_1^2 + vB_1^4) \\ \cdots \\ \kappa_{1,12}(A_1^6 + uA_1^3B_1^2 + vB_1^4) \\ \kappa_{1,13}(A_2^6 + uA_2^3B_2^2 + vB_2^4) \\ \cdots \\ \kappa_{1,20}(A_2^6 + uA_2^3B_2^2 + vB_2^4) \end{pmatrix}.$$

Given the A_i's and B_i's, we can solve for u and v, yielding a non-zero vector in the kernel of M, showing the system is under-determined.

Fortunately, we can circumvent this problem using the dual equations (18)

$$U'_\ell(-\ell\kappa_1, A^*, B^*)(\ell^4 A) = \mathcal{A}_\ell(-\ell\kappa_1, A^*, B^*), \quad U'_\ell(-\ell\kappa_1, A^*, B^*)(\ell^6 B) = \mathcal{B}_\ell(-\ell\kappa_1, A^*, B^*)$$

in Algorithm 11.

Note that the system is $O(\ell^2) \times O(\ell^2)$, leading to a $O(\ell^{2\omega})$ time complexity. Adapting this to the case of $\mathcal{B}_\ell$ is straightforward.

Algorithm 11: Computing $\mathcal{A}_\ell(X,Y,Z) \bmod p$.

Function ComputeAMod(ℓ, D, $H_D(z)$, p):
- **Input :** ℓ an odd prime, D the discriminant of an imaginary quadratic order $\mathcal{O}$ of discriminant D with class number $h(D) \geq \ell+2$; H_D is the class polynomial associated to order $\mathcal{O}$; p prime with $p \equiv 1 \bmod \ell$ and $4p = t^2 - \ell^2 v^2 D$, $v \not\equiv 0 \bmod \ell$
- **Output:** $\mathcal{A}_\ell(X,Y,Z) \bmod p$
1. $(\mathcal{E}_i, \{P_{ik}, \mathcal{E}'_{ik}\}_{1\leq k\leq \ell+1})_{1\leq i\leq h} \leftarrow$PartialVolcano($\ell$, D, $H_D(X)$, p)
2. Inject the values $(\kappa_{1,i}, A'_{ik}, B'_{ik})$ in equation (18) to get a linear system with enough rows
3. Solve the relevant linear system
4. **return** $\mathcal{A}_\ell$.

6. Computing the isogenous curve *à la* Atkin

The idea is to generalize the approach in [1, 2, 33], that is exploit q-series identities to get the parameters (κ, A^*, B^*), where we write κ for κ_1 from now on. As a matter of fact, we need relations involving $\tilde{E}_{2k} = E_{2k}(q^\ell)$, from which all other quantities follows: $A^* = -3\ell^4\tilde{E}_4$, $B^* = -2\ell^6\tilde{E}_6$.

6.1. Properties of U_ℓ.

We write for readability $U(\kappa, E_4, E_6) = U_\ell(X, E_4, E_6, \Delta)$ after replacing Δ by its expression and

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

and propagate the notation to double derivatives.

The polynomial U is homogeneous with weights, so that

$$(\ell+1)U = \kappa\partial_\kappa + 2E_4\partial_4 + 3E_6\partial_6. \tag{19}$$

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

$$\ell\partial_\kappa = \kappa\partial_{\kappa\kappa} + 2E_4\partial_{\kappa 4} + 3E_6\partial_{\kappa 6}, \tag{20}$$

$$(\ell-1)\partial_4 = \kappa\partial_{\kappa 4} + 2E_4\partial_{44} + 3E_6\partial_{46}, \tag{21}$$

$$(\ell-2)\partial_6 = \kappa\partial_{\kappa 6} + 2E_4\partial_{46} + 3E_6\partial_{66}. \tag{22}$$

6.2. Getting the isogenous curve from U_ℓ.

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

Proposition 6.1. *The value of $\tilde{E}_4$ is given by*

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

Proof: We differentiate (using (6)) $U(\kappa, E_4, E_6) = 0$ to get

$$\kappa'\partial_\kappa + E_4'\partial_4 + E_6'\partial_6 = 0. \tag{23}$$

We differentiate (15) leading to

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

Use (15) to replace $\ell\tilde{E}_2$ by $2\kappa/\ell + E_2$ to get

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

that we plug in (23) together with the expressions for ${E_4}'$ and ${E_6}'$ from equation (9) to get a polynomial of degree 1 in E_2 whose coefficient of E_2 is

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

which we recognize in (19). Therefore, we get

$$(\ell+1)UE_2 + \frac{\ell\,\partial_\kappa}{4}\left(\frac{4\kappa^2}{\ell^2} - (\ell^2\tilde{E}_4 - E_4)\right) - 2E_6\partial_4 - 3E_4^2\partial_6 = 0 \tag{24}$$

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

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

Proposition 6.2. *The value of $\tilde{E}_6$ may be written*

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

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

Proof: We differentiate (23).

$$\kappa''\partial_\kappa + \kappa'(\kappa'\partial_{\kappa\kappa} + E_4'\partial_{\kappa 4} + E_6'\partial_{\kappa 6}) \tag{25}$$

$$+\quad E_4''\partial_4 + E_4'(\kappa'\partial_{4\kappa} + E_4'\partial_{44} + E_6'\partial_{46}) \tag{26}$$

$$+\quad E_6''\partial_6 + E_6'(\kappa'\partial_{6\kappa} + E_4'\partial_{64} + E_6'\partial_{66}) = 0 \tag{27}$$

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

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

to be used in (25). Differentiating relations of (9), 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 (26) and (27) respectively. We replace $\tilde{E}_4$ by its value from (24), and $\tilde{E}_2$ using $\kappa = (\ell/2)(\ell\tilde{E}_2 - E_2)$. This finally yields an expression as polynomial in E_2:

$$C_2 E_2^2 + C_1 E_2 + C_0 = 0.$$

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

By luck(?)

Proposition 6.3. *The coefficients C_1 and C_2 vanish for a triplet such that $U_\ell(\kappa, E_4, E_6) = 0$.*

Sketch of the proof: The strategy to prove this is the same in both cases. Replace $\partial_{\kappa\kappa}$, ∂_{44} and ∂_{66} by their values from (20). Factoring the resulting expressions yields the same factor $\kappa\partial_\kappa + 2E_4\partial_4 + 3E_6\partial_6$, 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

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

where N is a polynomial in degree 3 in ℓ

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

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

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

6.2.3. *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 $\kappa = 584$ and compute

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

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

7. Implementation and numerical results

A lot of trials were done using Maple programs, some of which were then rewritten in Magma (version 2.26-10), for speed. See the author's web page. Computing the polynomials for $\ell \leq 100$ takes a few minutes on a classical laptop. Checking them is done using SEA, as mentioned in [35].

We give some examples of the *relative height* $\tilde{H}$ for some of our polynomials. Here $\tilde{H}(P) = H(P)/((\ell+1)\log\ell)$. Note that these quantities seem to stabilize when ℓ increases and are in accordance with Proposition 4.6.

ℓ	$\tilde{H}(\Phi_\ell^t)$	$\tilde{H}(\Phi_\ell^c)$	$\tilde{H}(\Phi_\ell^*)$	$\tilde{H}(U_\ell)$
2	15.72	4.00	−−	−−
3	11.14	1.51	−−	0.32
5	11.243	0.762	−−	0.526
7	9.787	0.582	−−	0.640
11	10.130	1.842	1.120	0.670
13	9.565	0.367	0.941	0.688
17	9.581	0.958	0.714	0.690
19	9.365	0.648	0.630	0.695
23	9.438	1.995	0.419	0.698
101	−−	1.111	0.159	0.778
103	−−	0.740	0.249	0.779
107	−−	2.218	0.228	0.781
109	−−	0.379	0.213	0.782

Data are computed using the polynomials available in MAGMA: Φ_ℓ^c is called *canonical polynomial* and Φ_ℓ^* is called *Atkin polynomial*. In the case of Φ_ℓ^c, the height depends on ℓ mod 12. Still, Atkin's minimal functions remain the best choice for large ℓ's.

8. Conclusions

We have given several methods for computing the Fricke and Charlap-Coley-Robbins polynomials, and manage to adapt known algorithms for this task. We also included representations as fractions in polynomials. In some cases, U_ℓ has smaller height, at long as ℓ is small.

In isogeny cryptography they are useful for relatively small ℓ's, if we store $(U_\ell, \mathcal{A}_\ell, \mathcal{B}_\ell)$. If one wants to compute an isogeny, it is enough to compute a root of U_ℓ followed by instantiations of three polynomials.

Also, we insisted on families of modular forms. Some of the techniques can be used for *ad hoc* forms.

In a follow up work [34], we look at modular polynomials for cuspidal η-products, as already described by Fricke, one of which was recommanded by Atkin to replace the U_ℓ polynomial when $\ell \equiv 11 \bmod 12$.

Acknowledgments. The author wishes to thank A. Bostan and F. Chyzak for helpful discussions around some aspects of this work; special thanks to the former for his impressive list of references for the fast evaluation of hypergeometric functions. Thanks also to L. De Feo for his updates on cryptographic applications of isogenies.

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] B. 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] B. C. Berndt, H. H. Chan, J. Sohn, and S. H. Son. Eisenstein series in Ramanujan's lost notebook. *Ramanujan J.*, 4(1):81–114, 2000.
[5] B. C. Berndt and A. J. Yee. Ramanujan's contributions to Eisenstein series, especially in his lost notebook. In *Number theoretic methods (Iizuka, 2001)*, volume 8 of *Dev. Math.*, pages 31–53. Kluwer Acad. Publ., Dordrecht, 2002.
[6] I. Blake, G. Seroussi, and N. Smart. *Elliptic curves in cryptography*, volume 265 of *London Math. Soc. Lecture Note Ser.* Cambridge University Press, 1999.
[7] J. M. Borwein and P. B. Borwein. A cubic counterpart of Jacobi's identity and the AGM. *Trans. Amer. Math. Soc.*, 323(2):691–701, 1991.
[8] A. Bostan, F. Morain, B. Salvy, and É. Schost. Fast algorithms for computing isogenies between elliptic curves. *Math. Comp.*, 77(263):1755–1778, 2008.
[9] R. Brent and P. Zimmermann. *Modern Computer Arithmetic*. Cambridge University Press, 2010.
[10] R. Bröker, K. E. Lauter, and A. V. Sutherland. Modular polynomials via isogeny volcanoes. *Math. Comput.*, 81(278):1201–1231, 2012.
[11] W. Castryck, T. Lange, C. Martindale, L. Panny, and J. Renes. CSIDH: an efficient post-quantum commutative group action. In T. Peyrin and S. 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.
[12] 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.
[13] D. Charles and K. Lauter. Computing modular polynomials. *LMS J. Comput. Math.*, 8:195–204, 2005.
[14] D. X. Charles, K. E. Lauter, and E. Z. Goren. Cryptographic hash functions from expander graphs. *J. Cryptol.*, 22(1):93–113, 2009.
[15] H. Cohen and F. Strömberg. *Modular forms – a classical approach*, volume 179 of *Graduate Studies in Mathematics*. American Mathematical Society, 2017.
[16] P. Cohen. On the coefficients of the transformation polynomials for the elliptic modular function. *Math. Proc. Cambridge Philos. Soc.*, 95:389–402, 1984.
[17] J.-M. Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291, 2006. `http://eprint.iacr.org/2006/291`.
[18] R. Dupont. Fast evaluation of modular functions using Newton iterations and the AGM. *Math. Comp.*, 80(275):1823–1847, 2011.
[19] A. El Basraoui and A. Sebbar. Zeros of the Eisenstein series E_2. *Proc. Amer. Math. Soc.*, 138(7):2289–2299, 2010.
[20] 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.
[21] A. Enge. Computing modular polynomials in quasi-linear time. *Math. Comp.*, 78(267):1809–1824, 2009.
[22] A. Enge, W. Hart, and F. Johansson. Short addition sequences for theta functions. *J. Integer Seq.*, 21(2):Art. 18.2.4, 34, 2018.
[23] A. Erdélyi, editor. *Higher transcendental functions*, volume II. McGraw-Hill, 1953.

[24] L. D. Feo, J. Kieffer, and B. Smith. Towards practical key exchange from ordinary isogeny graphs. In T. Peyrin and S. 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.

[25] L. D. Feo, D. Kohel, A. Leroux, C. Petit, and B. Wesolowski. Sqisign: Compact post-quantum signatures from quaternions and isogenies. In S. Moriai and H. 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.

[26] R. Fricke. *Die elliptischen Funktionen und ihre Anwendungen – Zweiter Teil : Die Algebraischen Ausführungen*. Teubner, Leipzig, 1922.

[27] D. Jao and L. De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In B. 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.

[28] F. Johansson. Computing hypergeometric functions rigorously. *ACM Trans. Math. Softw.*, 45(3):30, 2019.

[29] M. Kaneko and D. Zagier. Supersingular j-invariants, hypergeometric series, and Atkin's orthogonal polynomials. 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 97–126. American Mathematical Society, International Press, 1998.

[30] M. Kaneko and M. Koike. On modular forms arising from a differential equation of hypergeometric type. *Ramanujan J.*, 7(1-3):145–164, 2003. Rankin memorial issues.

[31] H. Labrande. Computing Jacobi's theta in quasi-linear time. *Math. Comp.*, 87(311):1479–1508, 2018.

[32] M. Mezzarobba and B. Salvy. Effective bounds for p-recursive sequences. *J. Symb. Comput.*, 45(10):1075–1096, 2010.

[33] 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.

[34] F. Morain. Using modular polynomials for eta products to compute isogenies. `https://inria.hal.science/hal-04423470`, January 2024. Preprint.

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

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

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

[38] R. A. Rankin. *Modular forms and functions*. Cambridge University Press, 1977.

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

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

[41] J.-P. Serre. *A course in arithmetic*, volume No. 7 of *Graduate Texts in Mathematics*. Springer-Verlag, New York-Heidelberg, 1973. Translated from the French.

[42] J.-P. Serre. Sur la lacunarité des puissances de η. *Glasgow Math. J.*, 27:203–221, 1985.

[43] A. V. Sutherland. On the evaluation of modular polynomials. In *ANTS X—Proceedings of the Tenth Algorithmic Number Theory Symposium*, volume 1 of *Open Book Ser.*, pages 531–555. Math. Sci. Publ., Berkeley, CA, 2013.

[44] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.2)*, 2020. `https://www.sagemath.org`.
[45] J. van der Hoeven. Fast evaluation of holonomic functions. *Theor. Comput. Sci.*, 210(1):199–215, 1999.
[46] J. van der Hoeven. Fast evaluation of holonomic functions near and in regular singularities. *J. Symb. Comput.*, 31(6):717–743, 2001.

Appendix A. A script to check the computations

This `SageMath` [44] 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 Fricke polynomials using the notations of the prep

# 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 Fricke 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()
```

Appendix B. Some values of Fricke/CCR polynomials

B.1. **Prime indices.** Note there is a sign flip compared to [35]: They use $B = 3E_6$, whereas we use $B = -3E_6$ which is coherent with Atkin's work, say.

For $\ell = 2$, $U_2(X) = X^3 + AX + B$ itself, which is the minimal polynomial of any of the 2-torsion points. For $\ell = 3$, we compute

$$W_3(X, E_4, E_6, \Delta) = X^4 + 1464E_6X^3 + (8760E_4^3 - 1185643008\Delta)X^2 + (17504E_4^3 - 152195991552\Delta)E_6X$$
$$+ 11664E_4^6 - 1790914074624\Delta E_4^3 - 20889728069861376\Delta^2;$$

Remark that $U_3 = \psi_3/3$. Also

$$\mathcal{A}_3(X, E_4, E_6, \Delta) = -252E_4X^3 - 720E_6X^2 - 684E_4^2X - 216E_4E_6$$

$$\mathcal{B}_3(X, E_4, E_6, \Delta) = -1464E_6X^3 - 4368E_4^2X^2 - 4344E_4E_6X - 1440E_4^3 + 746496\Delta$$

For $\ell = 5$:

$$\mathcal{A}_5 = -1890E_4X^5 - 18720E_6X^4 - 74160E_4^2X^3 - 146880E_4E_6X^2 + (-145440E_4^3 + 199065600\Delta)X - 57600E_4^2E_6,$$

$$\mathcal{B}_5 = -31260E_6X^5 - 312480E_4^2X^4 - 1249440E_4E_6X^3 + (-2497920E_4^3 + 763084800\Delta)X^2 - 2496960E_4^2E_6X - 998400E_4^4 + 1725235200\Delta E_4.$$

ℓ	$\tilde{H}(V_\ell)$	$\tilde{H}(W_\ell)$	$\tilde{H}(\mathcal{A}_\ell)$	$\tilde{H}(\mathcal{B}_\ell)$
5	3.266	4.336	1.063	1.268
7	3.050	4.207	0.973	1.167
11	2.939	3.979	0.896	1.016
13	2.856	3.969	0.864	0.983
17	2.770	3.883	0.831	0.919
19	2.754	3.831	0.820	0.901
23	2.723	3.764	0.799	0.869
101	2.471	3.527	0.801	0.818
103	2.469	3.517	0.801	0.818
107	2.466	3.516	0.803	0.819
109	2.467	3.515	0.803	0.819

B.2. **An example with $N = 6$.** Let $f = E_4$. We give some details for the computation of the modular polynomial for $E_4(q^6)$. For each representative $R = [a, b; c, d]$ of a coset, we compute the matrices $[Na, Nb; c, d] = UR'$ with $U \in \Gamma$ and $R' = [A, B; 0, D]$ to which corresponds the conjugate $f(R'\tau)$. Following Proposition 2.16, we group the conjugates matrices R' w.r.t. D. We find

R	U	R'	$f(R'\tau)$
$[1, 0; 0, 1]$	$[1, 0; 0, 1]$	$[6, 0; 0, 1]$	$1296 + O(z^{36})$
$[2, 1; 3, 2]$	$[1, -3; -1, 4]$	$[3, 0; 0, 2]$	$81 + 19440z^9 + O(z^{18})$
$[1, 0; 3, 1]$	$[0, 1; -1, 2]$	$[3, 1; 0, 2]$	$81 + 19440\zeta_6^3 z^9 + O(z^{18})$
$[1, 1; 2, 3]$	$[1, -2; -1, 3]$	$[2, 0; 0, 3]$	$16 + 3840z^4 + 34560z^8 + O(z^{12})$
$[1, 0; 2, 1]$	$[0, 1; -1, 3]$	$[2, 1; 0, 3]$	$16 + 3840\zeta_6^2 z^4 + 34560\zeta_6^4 z^8 + O(z^{12})$
$[1, 0; 4, 1]$	$[-1, 2; -2, 3]$	$[2, 2; 0, 3]$	$16 + 3840\zeta_6^4 z^4 + 34560\zeta_6^8 z^8 + O(z^{12})$
$[1, 5; 1, 6]$	$[1, -5; -1, 6]$	$[1, 0; 0, 6]$	$1 + 240z + 2160z^2 + 6720z^3 + 17520z^4 + O(z^5)$
$[1, 0; 1, 1]$	$[0, 1; -1, 6]$	$[1, 1; 0, 6]$	$1 + 240\zeta_6 z + 2160\zeta_6^2 z^2 + 6720\zeta_6^3 z^3 + 17520\zeta_6^4 z^4 + O(z^5)$
$[1, 1; 1, 2]$	$[0, 1; -1, 6]$	$[1, 2; 0, 6]$	$1 + 240\zeta_6^2 z + 2160\zeta_6^4 z^2 + 6720\zeta_6^6 z^3 + 17520\zeta_6^8 z^4 + O(z^5)$
$[1, 2; 1, 3]$	$[0, 1; -1, 6]$	$[1, 3; 0, 6]$	$1 + 240\zeta_6^3 z + 2160\zeta_6^6 z^2 + 6720\zeta_6^9 z^3 + 17520\zeta_6^{12} z^4 + O(z^5)$
$[3, 1; 5, 2]$	$[-3, 11; -5, 18]$	$[1, 4; 0, 6]$	$1 + 240\zeta_6^4 z + 2160\zeta_6^8 z^2 + 6720\zeta_6^{12} z^3 + 17520\zeta_6^{16} z^4 + O(z^5)$
$[1, 0; 5, 1]$	$[-4, 5; -5, 6]$	$[1, 5; 0, 6]$	$1 + 240\zeta_6^5 z + 2160\zeta_6^{10} z^2 + 6720\zeta_6^{15} z^3 + 17520\zeta_6^{20} z^4 + O(z^5)$

The corresponding power sums $S_{D,t}(f)$ have q-expansion in q^A (with $A = N/D$) and rational integer coefficients. For instance

D	A	$S_1(D)$
1	6	$1296 + 311040q^6 + O(q^{12})$
2	3	$162 + 349920q^3 + 2838240q^6 + O(q^9)$
3	2	$48 + 322560q^2 + 2903040q^4 + 8720640q^6 + O(q^8)$
6	1	$6 + 362880q + 2943360q^2 + 9810720q^3 + O(q^4)$

Summing all these, we get

$$S_1 = 1512 + 362880q + 3265920q^2 + 10160640q^3 + O(q^4)$$

in which we recognize $1512E_4$. Finally

$$\begin{aligned}
\Phi[E_4(6\tau)] &= X^{12} - 1512E_4X^{11} + 296316E_4^2X^{10} + 120(-181381E_4^3 + 3782160000\Delta)X^9 \\
&-270E_4(-2610581E_4^3 + 45664128000\Delta)X^8 - 72E_4^2(155634011E_4^3 + 97714341360000\Delta)X^7 \\
&+12(7370195077E_4^6 + 29670256575360000\Delta E_4^3 + 32018727707136000000\Delta^2)X^6 \\
&-1944E_4(170853343E_4^6 - 4897879320240000\Delta E_4^3 + 237438876026880000000\Delta^2)X^5 \\
&+45E_4^2(15102174661E_4^6 - 954640801014912000\Delta E_4^3 + 47160528043659264000000\Delta^2)X^4 \\
&+320(-2535407921E_4^9 - 1030763754097002000\Delta E_4^6 - 34720666413600499200000\Delta^2E_4^3 + 21192307848797184000000000\Delta^3)X^3 \\
&-373248E_4(-1520467E_4^9 - 246490368694140000\Delta E_4^6 - 9901962075946860000000\Delta^2E_4^3 + 12428563306452480000000000\Delta^3)X^2 \\
&+3547348992E_4^2(-61E_4^9 - 49957886310000\Delta E_4^6 - 1550463173247600000\Delta^2E_4^3 + 15137343021240000000000\Delta^3)X \\
&+34828517376(-E_4^3 + 54000\Delta)(-E_4^9 + 151013228706000\Delta E_4^6 - 224179462188000000\Delta^2E_4^3 + 1879994705688000000000\Delta^3).
\end{aligned}$$

Appendix C. Numerical data for the isogeny volcano algorithm

LIX - Laboratoire d'informatique de l'École polytechnique *and* GRACE - Inria Saclay–Île-de-France

Email address: morain@lix.polytechnique.fr

i	$\mathcal{E}_i = [A_i, B_i]$	$\mathcal{E}_i^*$	$\sigma(\mathcal{E}_i^*)$
1	$[1582, 902]$	$[594, 422]$	226
		$[1543, 911]$	1542
		$[937, 1244]$	1283
		$[1333, 561]$	1691
		$[879, 342]$	1212
		$[757, 1578]$	1290
2	$[1662, 405]$	$[1770, 433]$	529
		$[1439, 1411]$	1536
		$[259, 355]$	1810
		$[382, 1793]$	1733
		$[1472, 543]$	433
		$[413, 1603]$	1203
3	$[1451, 1331]$	$[1096, 1433]$	743
		$[1371, 1367]$	98
		$[1105, 1195]$	207
		$[1657, 1699]$	787
		$[811, 812]$	1769
		$[779, 1311]$	18
4	$[1013, 747]$	$[1691, 473]$	1705
		$[509, 342]$	1245
		$[1642, 417]$	1406
		$[127, 765]$	1519
		$[905, 1464]$	145
		$[1277, 254]$	1224
5	$[224, 753]$	$[1485, 892]$	1566
		$[823, 1106]$	908
		$[397, 1451]$	1729
		$[131, 673]$	450
		$[654, 1798]$	1353
		$[1805, 1025]$	1238
6	$[1128, 1504]$	$[1275, 1672]$	1176
		$[1409, 761]$	1362
		$[907, 1757]$	309
		$[824, 1267]$	781
		$[578, 1320]$	1208
		$[1168, 1207]$	597
7	$[91, 725]$	$[1184, 542]$	1284
		$[1753, 297]$	859
		$[1440, 1524]$	1268
		$[421, 410]$	517
		$[1626, 1013]$	245
		$[198, 159]$	1260

TABLE 1. Values for $\ell = 5$ and $p = 1811$.