On the Computation of Modular Polynomials for Elliptic Curves

Ian F. Blake[1], János A. Csirik[1], Michael Rubinstein[2⋆], and Gadiel Seroussi[1]

[1] Hewlett-Packard Laboratories,
1501 Page Mill Road, Palo Alto, CA 94304
ifblake, csirik, seroussi@hpl.hp.com
[2] Department of Mathematics
University of Texas at Austin
Austin, TX 78712-1082
miker@math.utexas.edu

Abstract. An essential aspect of the use of elliptic curves over a finite field in public key cryptosystems is to determine the precise order of the additive group of rational points of the curve. All known effective point-counting algorithms for such elliptic curves require the computation of modular polynomials. Several approaches to the computation of modular polynomials and variants of them have been considered in the literature. The purpose of this work is to give a unified treatment of these approaches, and to report on computational experience with their efficient implementations.

1 Introduction

Over the past decade there has been increasing interest in elliptic curve cryptosystems, to the point where they are now incorporated into several industry standards for public key cryptography. For many applications they appear to offer an attractive alternative to public key systems based on integer factorization or finite field discrete logarithms, since they allow for much lower key sizes for the same level of security, based on current knowledge of the best available algorithms for the respective underlying problems. These lower key sizes often translate to greater implementation efficiency to make elliptic curve systems particularly attractive for low power applications such as smart cards and wireless products [1].

Elliptic curve systems are based on the discrete logarithm problem on the additive group of rational points of an elliptic curve over a finite field, rather than, say, the multiplicative group of a finite field of integers modulo a prime. For cryptographic reasons, the group order is required to be divisible by a very large prime. Therefore, determining the precise number of rational points on the curve (the *point-counting* problem) is of fundamental importance. This computation can be challenging since the group orders of practical interest might be of the

⋆ Work done while at Hewlett-Packard Laboratories.

order of 2^{200} or larger. The most successful general algorithm for point counting to date is *Schoof's algorithm* ([12], [13]). The most practical version of this algorithm includes improvements due to Elkies and Atkin and is known as the *SEA algorithm*.

In this paper we will only consider elliptic curves over prime fields. Let p be a prime, $K = \mathbb{F}_p$ a finite field with p elements, $\overline{K}$ its algebraic closure, and E an elliptic curve over K. Let the short Weierstrass equation of E be

$$E: \ y^2 = x^3 + ax + b,$$

with j-invariant $j = 6912a^3/(4a^3 + 27b^2)$. We denote by $E(F)$ the set of points on E with coordinates in F, for any field F, $K \subseteq F$. We refer to $E(\overline{K})$ as the set of points of E, and to $E(K)$ as the set of *rational* points of E. It is known by Hasse's theorem that $|E(K)| = p + 1 - t$ where t is the *trace* of the Frobenius endomorphism, bounded by $|t| \leq 2\sqrt{p}$. Determining t is then equivalent to determining the group order. The essence of Schoof's algorithm is the determination of t modulo primes ℓ for $\ell \leq L$ where L is the smallest prime such that $\prod_{\ell \leq L} \ell > 4\sqrt{p}$. (In a typical application, L might be between 100 and 200.) It then follows from the Chinese Remainder Theorem (CRT) that the value of t can be recovered uniquely and the group order obtained. The number of primes needed is $O(\log p / \log\log p)$ and $L = O(\log p)$. Now we briefly outline Schoof's algorithm to determine the value of t modulo a prime ℓ. In order to avoid various mathematical subtleties, we will stipulate here that all elliptic curves considered will have j-invariant different from 0 or 1728 and will not be supersingular (i.e., $t \neq 0$). These conditions are easily tested for and are extremely rare for randomly chosen curves. We will also assume that $p > 2\ell + 2$, which holds in practical applications, to guarantee that certain inverses modulo ℓ exist.

The Frobenius endomorphism φ of E maps the point $P = (x, y) \in E(\overline{K})$ to (x^p, y^p). It satisfies the Frobenius equation $\varphi^2(P) - [t]\varphi(P) + [p]P = \mathcal{O}$, or

$$(x^{p^2}, y^{p^2}) + [p](x, y) = [t](x^p, y^p). \tag{1}$$

Here, for integers $m \geq 0$ and curve points P, $[m]P = P + P + \cdots + P$ (m times) and $\mathcal{O}$ is the point at infinity, the neutral element of the group. Restricting P to $E[\ell]$, the set of ℓ-torsion points of E, both p and t are reduced modulo ℓ in (1). We can compute the left hand side of (1) by using the point addition formulae and then compare it to the right hand side as t runs through the values modulo ℓ. The equation will have a solution for exactly one value of t mod ℓ. We calculate with (1) as though x and y were indeterminates, subject only to the equation of the curve and the relation $\phi_\ell(x) = 0$, where the ℓth *division polynomial* ϕ_ℓ vanishes exactly at the x-coordinates of points in $E[\ell]$ (see, e.g., [1]). The degree of ϕ_ℓ is $(\ell^2 - 1)/2$ when ℓ is an odd prime. For large ℓ, the high degree of this polynomial makes the computations cumbersome. The contributions of Elkies and Atkin allow us in some cases to replace this polynomial by a so-called *kernel polynomial*, of degree $(\ell - 1)/2$.

The simplification is achieved when the discriminant of the Frobenius equation, $\Delta_t = t^2 - 4p$, is a square modulo ℓ, in which case ℓ is referred to as an *Elkies prime*. Note that Δ_t can never be the square of an integer, so it will be a square modulo about half of the primes ℓ. The prime ℓ is an Elkies prime if and only if there is another elliptic curve E', also defined over $\mathbb{F}_p$, such that E and E' are isogenous via an isogeny of degree ℓ. The kernel of this ℓ-isogeny is a subgroup C of order ℓ of E. Then the kernel polynomial $h_{E'}(x)$, which vanishes exactly at the x-coordinates of points in C, and hence is of degree $(\ell-1)/2$, can be used in place of ϕ_ℓ above. The use of this polynomial, rather than the division polynomial, leads to a factor of improvement in the overall running time of the algorithm of as much as $\ell^2 = O(\log^2 p)$ [7, 1].

The primes ℓ for which Δ_t modulo ℓ is not a square are referred to as *Atkin primes*. If ℓ is an Atkin prime then t modulo ℓ cannot be determined as above, but it can be restricted to lie in a proper subset of $\{0, 1, \ldots, \ell-1\}$. We discuss this case in Section 4.

Since t modulo ℓ is unknown at first, we cannot determine directly whether Δ_t is a square modulo ℓ. Instead, we will try to find an ℓ-isogenous curve E', if one exists. The ℓ-th classical *modular polynomial* $\Phi_\ell(X, Y)$ is a symmetric bivariate polynomial with integer coefficients, and of degree $\ell+1$ in each variable, with the property that the $\ell+1$ curves ℓ-isogenous to E have j-invariants which are roots of $\Phi_\ell(X, j) = 0$. The sought after E' therefore exists if and only if $\Phi_\ell(X, j) = 0$ has a root $j' \in \mathbb{F}_p$. (see, e.g., [12, 7, 6]). Thus, a critical aspect of all known effective point counting algorithms is the determination of modular polynomials from which the j-invariant of an ℓ-isogenous curve may be obtained, and thence the kernel polynomial.

Classical modular polynomials are often computed over $\mathbb{Z}$, stored, and then reduced modulo desired primes p as needed. The computation, storage and manipulation of these polynomials is challenging due to the large number of coefficients and their sizes. For this reason, variants with fewer and smaller coefficients have been proposed, which can still be used to obtain the information supplied by the classical polynomials.

Note that for a given j, once j' is found by means of modular polynomials, it is relatively simple to determine the polynomial $h_{E'}(x)$, which is what is needed for the use of Elkies primes in the SEA algorithm. This involves taking various partial derivatives of whichever modular polynomials are being used. The details, which differ for the various modular polynomials, have been worked out in [12, 7, 10, 6] and will not be discussed here.

A unified framework for the computation of classical modular polynomials and a class of variants is presented in Section 2, where the difficulties involved and the techniques used are described, and the asymptotic complexity of the computation is analyzed. Sections 3 and 4 discuss two specific variants, due to Müller and Atkin, respectively. These sections attempt to unify the theoretical basis for these variants, an aspect of the problem that has been difficult to obtain from the literature. Finally, Section 5 contains a summary of the data collected on the computation of the three types of modular polynomials, providing persuasive

evidence of the advantages of using the Atkin approach to modular polynomials in the point-counting problem.

2 General approach to computing modular polynomials

Although our interest is in elliptic curves over finite fields, we will present the theory over the field of complex numbers. The justification that everything works the same way over finite fields is beyond the scope of this article and can be found in [13, 7] and the references contained therein. Observe that the arithmetic operations described in the algorithms of this paper can be carried out in any field $\mathbb{F}_p$ as long as $p > 2\ell + 2$.

Recall that any elliptic curve E defined over the complex numbers can be obtained as a quotient of $\mathbb{C}$ by a rank 2 lattice $\mathcal{L}$. By scaling if necessary, we may assume that $\mathcal{L}$ is spanned by 1 and some τ that is in the upper half plane. Therefore, the j-invariant of E can be considered as a function of a variable τ in the upper half plane. Letting $\mathrm{SL}_2(\mathbb{Z})$ (the group of 2-by-2 matrices with integer entries and determinant 1) act on the upper half plane via fractional linear transformations (i.e., $\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)\tau = (a\tau + b)/(c\tau + d)$), we see that the quotients of $\mathbb{C}$ by the lattices $(1, \tau)$ and $(1, \gamma\tau)$ are isomorphic elliptic curves for any τ in the upper half plane and any $\gamma \in \mathrm{SL}_2(\mathbb{Z})$, and therefore $j(\tau) = j(\gamma\tau)$. In particular, taking $\gamma = \left(\begin{smallmatrix} 0 & 1 \\ -1 & 0 \end{smallmatrix}\right) \in \mathrm{SL}_2(\mathbb{Z})$, we obtain the important fact that $j(\tau) = j(-1/\tau)$. Similarly, for $\gamma = \left(\begin{smallmatrix} 1 & 1 \\ 0 & 1 \end{smallmatrix}\right)$, we obtain that $j(\tau) = j(\tau + 1)$, so j has a Fourier expansion in $q = \exp(2\pi i\tau)$ as follows

$$j(q) = \frac{1}{q} + 744 + \sum_{n=1}^{\infty} c_n q^n, \qquad c_n \in \mathbb{Z}.$$

Many of the functions used in the development will be power series in q, while the ℓ-isogenous j-invariants are more conveniently described in terms of functions of the τ related to the given curve. Because of this it will often be convenient to use q and τ interchangeably as arguments of functions, e.g., the j-invariant function might be referred to as either $j(q)$ or $j(\tau)$. It can be shown the j-invariants of the $\ell + 1$ curves ℓ-isogenous to the given curve are precisely

$$j(\ell\tau),\ j(\frac{\tau + k}{\ell}),\ \text{for } k = 0,\ 1, \ldots, \ell - 1.$$

The ℓth classical modular polynomial $\Phi_\ell(X, Y)$ may be defined as follows. Consider first the univariate polynomial

$$\Phi_\ell(X, j(\tau))) = (X - j(\ell\tau)) \prod_{k=0}^{\ell-1} (X - j(\frac{\tau + k}{\ell})). \tag{2}$$

For a given curve with j-invariant $j(\tau)$, $\Phi_\ell(X, j(\tau))$ is the polynomial whose roots comprise the j-invariants of all the curves ℓ-isogenous to the given curve.

Considering now j as a formal q-series rather than a specific value, it can be shown that Φ_ℓ is irreducible over $\mathbb{C}[j(q)]$ and has coefficients in $\mathbb{Z}[j(q)]$ (see also Section 4). Hence, we can write

$$\Phi_\ell(X, j(q)) = \sum_{a,b} f_{ab} X^a j(q)^b, \quad \text{or} \quad \Phi_\ell(X, Y) = \sum_{a,b} f_{ab} X^a Y^b, \ f_{ab} \in \mathbb{Z}. \quad (3)$$

These polynomials can be subsumed under the following general formulation that includes the approaches of Müller and Atkin considered in the following sections. Let $f_1(q)$ and $f_2(q)$ be q-expansions such that there is a polynomial $G(X, Y) \in \mathbb{Z}[X, Y]$ such that

$$G(X, j(q)) = (X - f_1(\tau)) \prod_{k=0}^{\ell-1} \left(X - f_2(\frac{\tau + k}{\ell}) \right) = \sum_{a,b} g_{ab} X^a j(q)^b. \quad (4)$$

Our goal then is to determine G given sufficiently many terms from the q-expansions of f_1 and f_2. Let $-v_i$ denote the smallest exponent of q that occurs in the q-expansion of f_i. In each variant, v_1 and v_2 will be $O(\ell)$. For simplicity, we assume v_2 is positive, which happens to be true in each of our cases of interest.

This setup reduces to the calculation of the classical polynomial Φ_ℓ upon setting $f_1(q) = j(q^\ell)$ and $f_2(q) = j(q)$. Then we have $v_1 = \ell$ and $v_2 = 1$.

Our approach to determining the coefficients g_{ab} of G is to substitute the q-expansions for f_1 and f_2 into (4). Then, since the coefficient of each power of X in (4) is a polynomial in j, we can directly read off G using only coefficients of non-positive powers of q.

Our calculation is most effectively carried out by rewriting expression (4) in the form

$$G(X, j(q)) = (X - f_1(q)) X^\ell \exp\left(- \sum_{k=0}^{\ell-1} \sum_{m=1}^{\infty} \frac{f_2^m(q^{1/\ell} \exp(2\pi i k/\ell))}{m X^m} \right) \quad (5)$$

by expressing $(1 - u)$ as $\exp(\log(1 - u))$ and expanding the logarithm function.

Before getting into the details of the calculation, we note that although complex analysis is used to derive some of the relations involved, the actual computations only involve power series with integer coefficients. In the two situations of interest to us, it will make sense to actually carry out these computations modulo suitable prime numbers. If we want to calculate the number of points of elliptic curves over a fixed finite field, then it suffices to know the reduction of our modular polynomials to the same field. If, however, we wish to compute the polynomials over $\mathbb{Z}$ to store and use later for elliptic curves over varying finite fields, then for each of our variants the numbers involved (for example, the coefficients of G) grow at least exponentially in ℓ as ℓ increases. It is computationally more efficient, therefore, to calculate them modulo various primes and finally use the CRT to reconstruct them over $\mathbb{Z}$. Therefore, in the sequel, we assume that the computations are carried out over the field $\mathbb{F}_r$, where r is a prime larger than $\ell + 1$ (since inverses of integers up to $\ell + 1$ are required). We will discuss later the problem of reconstructing G over $\mathbb{Z}$.

Step 1. From the q-expansion of f_2 we can compute $f_2^m(q) = \sum_{i=-mv_2}^{\infty} \alpha(i,m)q^i$, and therefore

$$s_m = \sum_{k=0}^{\ell-1} f_2^m(q^{1/\ell}\exp(2\pi ik/\ell)) = \ell \sum_{n=-\lfloor mv_2/\ell \rfloor}^{\infty} \alpha(\ell n, m)q^n. \tag{6}$$

Now we can rewrite (5) as

$$G(X, j(q)) = (X - f_1(q))X^{\ell} \exp\left(-\ell \sum_{m=1}^{\infty} \sum_{n=-\lfloor mv_2/\ell \rfloor}^{\infty} \frac{\alpha(\ell n, m)q^n}{mX^m}\right). \tag{7}$$

Notice that since we know that $G(X, j(q))$ only contains non-negative powers of X, it suffices for us to only consider $m \leq \ell+1$ in the summation above. Similarly, since we only need non-positive powers of q to determine the coefficients of G, using the restriction already obtained on m we may restrict to $n \leq v_1 + (\ell + 1)v_2/\ell = O(\ell)$.

To obtain each of the required coefficients $\alpha(i,m)$, we need to calculate each of the f_2^m (where $m = 2, \ldots, \ell+1$), to $\ell(v_1 + 2v_2) + 2(v_2 + 1) = O(\ell^2)$ terms. Using a fast (FFT-based) method to carry out polynomial multiplications, we need $O(\ell^3 \log \ell)$ field operations over $\mathbb{F}_r$ to carry out Step 1.

Step 2. To evaluate the exponential in (7), we used an algorithm due to Brent and Kung [3] for the computation of the exponentiation of the univariate power series. This was also adapted for bivariate power series.

Let $B^{(m)} = \{-\lfloor mv_2/\ell \rfloor, -\lfloor mv_2/\ell \rfloor + 1, \ldots, v_1 + (\ell+1)v_2/\ell\}$ the set over which n needs to be summed in (7). Splitting $B^{(m)}$ into the negative, zero and positive subsets $B_-^{(m)}, B_0^{(m)}, B_+^{(m)}$, and using the notation $A_{n,m} = -\ell\alpha(\ell n, m)/m$, we can rewrite the exponential term in (7) as

$$\exp\left(\sum_{m=1}^{\ell+1} \sum_{n \in B_-^{(m)}} \frac{A_{n,m}q^n}{X^m}\right) \exp\left(\sum_{m=1}^{\ell+1} \frac{A_{0,m}}{X^m}\right) \exp\left(\sum_{m=1}^{\ell+1} \sum_{n \in B_+^{(m)}} \frac{A_{n,m}q^n}{X^m}\right). \tag{8}$$

Each exponential only needs to be evaluated to the $X^{-\ell-1}$ term, and the first and third can also be truncated after $O(\ell)$ powers of q. Therefore, our exponentiation algorithm can provide the answers in $O(\ell^2 \log \ell)$ operations. Since the answers can be multiplied out using FFT in no more than that many operations, we conclude that Step 2 requires $O(\ell^2 \log \ell)$ field operations in $\mathbb{F}_r$.

Step 3. To actually read off the coefficients of G from (4) multiplied out with the expansions of f_1 and f_2 already substituted, we need to invert a square matrix of size $v_2 + \max(0, v_1) = O(\ell)$ (the coefficients of these matrix come from coefficients of non-positive powers of q in $j(q)$, $j(q)^2$, etc., arising from (4)). Then we need to multiply this inverse with $O(\ell)$ different column vectors. Both steps take $O(\ell^3)$ arithmetic operations over $\mathbb{F}_r$ (or faster, but with methods that are seldom used in practice for the sizes of interest), and hence this is the overall complexity of Step 3.

We have now seen that we can calculate the coefficients of G modulo r in $O(\ell^{3+\epsilon})$ field operations in $\mathbb{F}_r$. Since each field operation takes $O(\log r \log\log r)$ (binary) operations, when using fast methods for multiplication, the overall complexity of obtaining G modulo r works out to $O(\ell^{3+\epsilon} \log r \log\log r)$.

The computation of $G(X,Y)$ can also be approached by using Newton's identities to determine the coefficients of $\prod_{k=0}^{\ell-1}(X - f_2(\frac{\tau+k}{\ell}))$ from the power sums s_m of (6). This approach yields a slightly higher asymptotic complexity, but in practice we found running times similar for both approaches.

To obtain the modular polynomials over $\mathbb{Z}$, we compute their reductions modulo sufficiently many primes r, and then reconstruct the integer coefficients using an efficient implementation of the CRT (Garner's algorithm [4]). If we take r to be of fixed size (for example, one machine word, to exploit fast machine arithmetic), then, as we will see in (9), the number of primes needed is $O(\ell \log \ell)$, which brings the total cost of the computation to $O(\ell^{4+\epsilon})$. If, on the other hand, we use "large" primes r, i.e., $\log r \approx \ell$, then the number of primes is $O(\log \ell)$, but the operations in $\mathbb{F}_r$ cost $O(\ell \log \ell)$ bit operations, which, again, brings the total cost to $O(\ell^{4+\epsilon})$. Recall that when the modular polynomials are computed directly over the (fixed) field of definition $\mathbb{F}_p$ of the elliptic curve, we have $\ell = O(\log p)$.

In the classical case, $f_2(q) = j(q)$ does not depend on ℓ. If we wish to compute modular polynomials for all primes $\ell \le L$, we can compute $f_2^m(q)$, $2 \le m \le L+1$ and use the results for all ℓ. Thus, the computational cost of Step 1 is amortized, and it reduces, for each modular polynomial, to $O(\ell^2)$ field operations over $\mathbb{F}_r$. The overall computation is still dominated by the matrix multiplication in Step 3, which cannot be amortized.

The ℓth classical modular polynomial has $(\ell^2+3\ell+4)/2$ coefficients (counting $f_{ab} = f_{ba}$ only once). By [5], the largest coefficient over $\mathbb{Z}$ has magnitude

$$\exp\left(6\ell(\log \ell + O(1))\right). \tag{9}$$

For example, for $\ell = 197$, the number of coefficients is 19702, the largest coefficient has 3724 decimal digits, and the total number of decimal digits in all coefficients exceeds 53 million. These magnitudes make classical modular polynomials difficult to compute, store, and manipulate, and have led researchers to look for more manageable variants. These variants will be obtained by using different q-series f_1 and f_2, which produce polynomials that provide similar information but are easier to compute and of smaller size. Two of these variants are discussed in the next section.

3 Variants

Let $\Gamma_0(\ell)$ be the set of those matrices in $\mathrm{SL}_2(\mathbb{Z})$ whose lower left entry is divisible by ℓ, and let g be a holomorphic function on the upper half plane that is invariant under the transformations in $\Gamma_0(\ell)$. We also require that g be meromorphic at the cusps and that its q-expansion around $i\infty$ have only integer coefficients. By

[11, V, Thm. 1],

$$G_g(X) = (X - g(\tau)) \prod_{k=0}^{\ell-1} (X - g(-1/(\tau + k))$$

is a polynomial over $\mathbb{Z}[j(\tau)]$. (The various expressions plugged into g here are just a complete set of coset representatives for $\Gamma_0(\ell)$ in $\mathrm{SL}_2(\mathbb{Z})$ acting on τ.) Observe that if the elliptic curve E is isogenous to an elliptic curve E' via an ℓ-isogeny, and the j-invariant of E is equal to $j(\tau)$, then the j-invariant of E' is going to be equal to $j(\ell\tau')$, where τ' is one of $\{\tau, -1/\tau, -1/(\tau+1), \ldots, -1/(\tau+\ell-1)\}$. (To verify this claim, refer to (2) and recall that $j(\tau) = j(-1/\tau)$ for any τ in the upper half plane.)

Therefore, if we have an elliptic curve E with j-invariant $j(E)$, we can set $j(\tau)$ in the coefficients of G_g to equal $j(E)$, to obtain a polynomial $\overline{G_g}$ in $\mathbb{C}[X]$. For any root $g(\tau')$ of $\overline{G_g}$ we can then find $j(\ell\tau')$ by using special properties of g. We shall now survey the various functions g that have been proposed.

The classical case. Take $g(\tau) = j(\ell\tau)$. This function is invariant under the action of $\Gamma_0(\ell)$ by [11, VI, Thm. 12] and [11, remark below (14')]. In this case, $g(\tau')$ equal to $j(\ell\tau')$, by definition of g. This modular polynomial is indeed the classical one, since

$$G_g(X) = (X - j(\ell\tau)) \prod_{k=0}^{\ell-1} (X - j(-\ell/(\tau+k)) = (X - j(\ell\tau)) \prod_{k=0}^{\ell-1} (X - j((\tau+k)/\ell),$$

which matches the definition of Φ_ℓ given before. (We used $j(\tau) = j(-1/(\ell\tau))$ again.) The kernel polynomial can now be determined from $j(\tau)$ and $j(\ell\tau)$ by following the procedure given in [13, §7].

A g given by a simple formula. In [10, Section 5.1] Müller proposed considering the $\Gamma_0(\ell)$-invariant function $g(\tau) = \ell^s(\eta(\ell\tau)/\eta(\tau))^{2s}$, where s is the smallest positive integer such that $v = s(\ell-1)/12$ is an integer and $\eta(\tau)$ is the Dedekind η function, $\eta(\tau) = q^{1/24}\prod_{n=1}^{\infty}(1-q^n)$. It is then also true that $g(-1/\ell\tau)$ is equal to $\ell^s/g(\tau)$. In this case,

$$G_g(X) = (X - g(\tau)) \prod_{k=0}^{\ell-1} (X - g(-1/(\tau+k)) = (X - g(\tau)) \prod_{k=0}^{\ell-1} (X - f((\tau+k)/\ell),$$

so we can see that the modular polynomial G_g can be calculated by the methods of the previous section, by letting $f_1(q) = g(q)$ and $f_2(q) = f(q)$ (therefore $v_1 = -v$ and $v_2 = v$). The ℓth polynomial in this case has $2 + (\ell+1)(v+1)/2$ coefficients. Notice that v can vary from $(\ell-1)/12$ to $(\ell-1)/2$, depending on the residue class of ℓ modulo 12. This will also affect the size of the largest coefficient, as can be seen on Figure 1 in Section 5.

Here $g(\tau')$ does not equal $j(\ell\tau')$, however the explicit formula for g allows us to determine $j(\ell\tau')$ from $g(\tau')$. For the details of this calculation, as well as the determination of the kernel polynomial from $j(\tau)$, $j(\ell\tau)$ and $f(\tau)$, refer to [10, Section 6.3].

A g invariant under $\tau \mapsto -1/(\ell\tau)$. If, in addition to all the conditions above, our function g is also invariant under the Atkin-Lehner involution, $\tau \mapsto -1/(\ell\tau)$, then we can proceed as follows. We can determine the modular polynomial by the previous methods, since

$$G_g(X) = (X - g(\tau)) \prod_{k=0}^{\ell-1} (X - g(-1/(\tau+k)) = (X - g(\tau)) \prod_{k=0}^{\ell-1} (X - g((\tau+k)/\ell).$$

Therefore, to obtain G_g we use the method of Section 2 with the q-series $f_1 = f_2 = g$.

Getting from $g(\tau')$ to $j(\ell\tau')$ is as follows. The polynomial $G_g(G, J)$ is satisfied by the pair $(j(\tau'), g(\tau'))$. Using the Atkin-Lehner involution, we see that $G_g(G, J)$ is also satisfied by the pair $(j(\ell\tau'), g(\tau'))$. Hence, for each value of $g(\tau')$, just find the roots of $G_g(X, g(\tau'))$ to obtain all possible values of $j(\ell\tau')$. We might obtain some extraneous pairs $(j(\tau), j(\ell\tau))$ that do not correspond to isogenies, but those will be eliminated when we try to calculate the corresponding kernel polynomial, since the extra polynomials obtained will easily be confirmed as invalid. The calculation of the kernel polynomial form the data given here is explained in [9], [6], [10, Section 6.4].

The description of how one would obtain such a g is beyond the scope of this article, but is described in [10, Section 5.3] and [7, 9, 6]. It is clear from Section 2 that we want a g with a pole of low order v at $i\infty$, since in this case $v_1 = v_2 = v$. Müller finds very good functions using the action of the Hecke operators on the weight 1 modular form $\eta(\tau)\eta(\ell\tau)$. Atkin (conjecturally) finds the best possible ones by lifting functions from the special fiber of the moduli scheme $X_0^+(\ell)/\mathbb{Z}_\ell$. This guarantees that $v < (\ell+31)/24$. The ℓth polynomial in this case has $(3v+1)(\ell+1)/2+2$ coefficients. Hence, the number of coefficients is no more than $\ell^2/16 + 5\ell/2 + 71/16$.

4 Atkin primes

When calculating with the classical modular polynomial, we can glean some information about t modulo ℓ even if ℓ is not an Elkies prime. In that case $\Phi_\ell(X, j)$ will split into factors of degree s and we can restrict t modulo ℓ to lie in one of no more than $2\phi(s)$ residue classes, namely $t^2 \equiv p(\zeta + \zeta^{-1} + 2) \pmod{\ell}$, for some primitive sth root of unity ζ in $\overline{\mathbb{F}_\ell}$. For a proof, see [13, Prop. 6.2] and [1].

It turns out that the splitting type of our variant modular polynomials is the same as that of Ψ_ℓ, so they can be used just as well for Atkin primes. The proof of this fact is based on the following theorem:

Theorem 1. *Let p be an odd prime, and let $R = \mathbb{Z}[j]$ (where j is a transcendental quantity over $\mathbb{Z}$). Let Q be the field of fractions of R. Fix a reduction homomorphism $R \to \mathbb{F}$, where $\mathbb{F}$ is a finite field of characteristic p. Let F be a polynomial over R such that neither F nor its reduction to $\mathbb{F}$ have multiple*

roots. Let G be a polynomial over R with no multiple roots. If F and G generate the same splitting field, then the reductions of F and G over $\mathbb{F}$ have the same splitting type.

For a proof, consult [10, Section 4.7] or [2].

Here F is going to be the classical modular polynomial, and G will be any of our variant modular polynomials. The conditions are verified as follows. By [10, Lemma 4.28], $j(\ell\tau)$ has invariance group $\Gamma_0(\ell)$. By [10, bottom of p. 59], for any of our variants, g has invariance group $\Gamma_0(\ell)$ (since g and f have the same invariance group). Therefore, by [10, Satz 4.32], the splitting field for both F and G is the one corresponding to the same group $\Gamma^*(\ell)$ (which is also defined in [10]). Finally, by [11, VI, Theorems 5 and 1], the roots of F and G are distinct, since both polynomials are irreducible over a field of characteristic zero).

5 Experimental results

Section 3 presented variants of the classical modular polynomials, due to Müller and Atkin. Although the various complexity measures of interest for these variants (computation time, coefficient size, number of coefficients) are asymptotically similar to the classical case, the smaller value of $v = \max\{|v_1|, v_2\}$ for the variants leads to complexities that are significantly lower in practice. Table 1 shows various parameters for the example of $\ell = 197$ mentioned at the end of Section 2. Figure 1, on the other hand, plots the number of decimal digits of the largest coefficient of the ℓth modular polynomial as a function of ℓ for the classical case, the first Müller variant [10, Section 5.1], and the Atkin variant. A similar plot for Atkin polynomials up to $\ell = 757$ also showed "logarithmic" growth, but the length of the largest coefficient for $\ell = 757$ is only 509 digits. Table 1 and Figure 1 clearly show the advantages of the Atkin approach. As for practical running times, in our experiments, obtaining an Atkin polynomial for ℓ around 200 took a few minutes, while obtaining a classical polynomial of the same order took several hours, both on similar computation platforms.

In fact, a recent implementation based on Atkin's approach [6] incorporates the computation of the modular polynomial "on the fly" as part of the point-counting procedure, allowing for efficient generation of cryptographically strong elliptic curves on limited-memory devices. This "on the fly" computation would be very difficult for classical modular polynomials, where the computation of Φ_ℓ would overwhelmingly dominate the point-counting procedure.

References

1. I.F. Blake, G. Seroussi and N.P. Smart. *Elliptic Curves in Cryptography.* Cambridge University Press, July, 1999.
2. I.F. Blake, J.A. Csirik, M. Rubinstein and G. Seroussi. On the computation of modular polynomials for elliptic curves, *HP Laboratories Technical Report*, to appear.

Table 1. Modular polynomial parameters for $\ell = 197$.

	v	Number of coefficients	Digits in largest coeff.	Total digits
Classical	197	19 702	3 724	53 431 792
Müller$_1$	47	4 754	516	1 560 448
Atkin	6	1 883	119	148 733

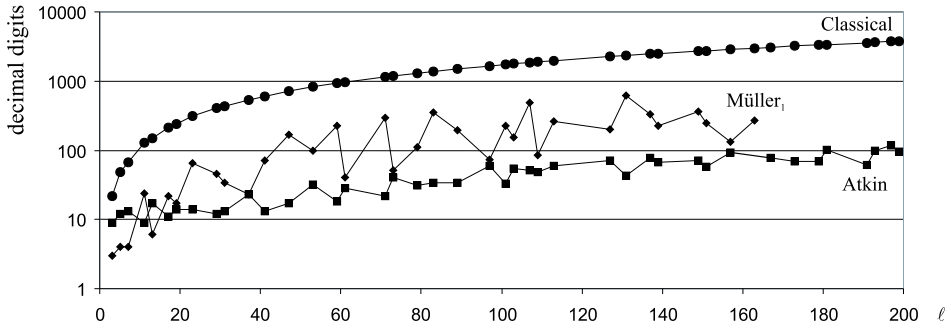

Fig. 1. Size of largest coefficient as a function of ℓ.

3. R.P. Brent and H.T. Kung. Fast algorithms for manipulating formal power series. *Jour. Assoc. Comp. Mach.*, **25**, 581–595, 1978.
4. H. Cohen. *A Course In Computational Algebraic Number Theory*. Springer-Verlag, GTM 138, 1993.
5. P. Cohen. On the coefficients of the transformation polynomials for the elliptic modular function. *Math. Proc. Camb. Phil. Soc.,* **95**, 389–402, 1984.
6. J.A. Csirik. Counting the points on an elliptic curve on a low memory device. Preprint, October, 1998.
7. 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*, American Mathematical Society International Press, **7**, 1998 21–76..
8. F. Lehmann, M. Maurer, V. Müller and V. Shoup. Counting the number of points on elliptic curves over finite fields of characteristic greater than three. In ANTS I, LNCS **877**, 60–70, 1991.
9. F. Morain. Calcul du nombre de points sur une courbe elliptique dans un corps fini: aspects algorithmiques. *J. Théorie des Nombres de Bordeaux*, **7**, 255–282, 1995.
10. V. Müller. Ein Algorithmus zur Bestimmung der Punktzahl elliptischer Kurven über endlichen Körpern der Charakteristik grösser drei. Ph.D. Thesis, Universität des Saarlandes, 1995.
11. B. Schoenberg. Elliptic Modular Functions: An Introduction. Springer-Verlag, 1974.

12. R. Schoof. Elliptic curves over finite fields and the computation of square roots mod p. *Math. Comp.*, **44**, 483–494, 1985.
13. R. Schoof. Counting points on elliptic curves over finite fields. *J. Théorie des Nombres de Bordeaux*, **7**, 219–254, 1995
14. J.H. Silverman. *Advanced Topics in the Arithmetic of Elliptic Curves*. Springer-Verlag, GTM 151, 1994.