1 Computing L-functions of Weil restrictions of elliptic curves over number fields

UROP Summer 2021 Project Writeup by Daniel Hu, supervised by Dr. Andrew Sutherland

§1.1 L-polynomials and L-functions background

Let $A/\mathbb{F}_q$ be an abelian variety of dimension g over a finite field of characteristic p. For prime $\ell \neq p$, the ℓ-adic Tate module [2, 7.5.5] is a free $\mathbb{Z}_\ell$-module of rank $2g$ defined as

$$T_\ell(A) := \varprojlim_i A[\ell^i]$$

The Weil polynomial is the characteristic polynomial of the Frobenius endomorphism acting on $T_\ell(A)$, and the **L-polynomial** is the reverse of the Weil polynomial. The L-polynomial $L_q(T)$ is an integer polynomial of degree $2g$ independent of ℓ. It is also the numerator of the zeta function of A, which is defined as the exponential generating function

$$Z_A(T) = \exp\left(\sum_{n=1}^{\infty} \frac{\#A(\mathbb{F}_{q^n})T^n}{n}\right)$$

By the Weil conjectures [1], this is a rational function and thus its numerator is well-defined. The L-polynomial possesses the following properties:

1. $L_q(T) = \sum_{i=0}^{2g} a_i T^i$ satisfies $a_0 = 1$ and $a_{2g-i} = q^{g-i} a_i$.
2. The complex roots of $L_q(T)$ all have absolute value $1/\sqrt{q}$.

The second condition implies the bounds $|a_i| \leq q^{i/2}\binom{2g}{i}$ for $1 \leq i \leq g$.

Let A/K be an abelian variety over a number field. Its L-function [2, 5.7.7] is given by

$$L(A, s) = \prod_{\mathfrak{p}} L_{\mathfrak{p}}(N(\mathfrak{p})^{-s})^{-1}$$

where $\mathfrak{p}$ ranges over all primes in $\mathcal{O}_K$. For all primes $\mathfrak{p}$ of good reduction, $L_{\mathfrak{p}}(T)$ is the L-polynomial of the reduction of A to the residue field $\mathcal{O}_K/\mathfrak{p}$. When $\mathfrak{p}$ is of bad reduction, there is still a polynomial $L_{\mathfrak{p}}(T)$, but with degree less than $2g$.

In the case when A is an elliptic curve, for good primes $\mathfrak{p}$ we define $a_{\mathfrak{p}}$ to be the trace of Frobenius of the reduction of A to $\mathcal{O}_K/\mathfrak{p}$, and we have

$$L_{\mathfrak{p}}(T) = 1 - a_{\mathfrak{p}}T + N(\mathfrak{p})T^2$$

For bad primes $\mathfrak{p}$, we have

$$L_{\mathfrak{p}}(T) = 1 - a_{\mathfrak{p}}T, \quad a_{\mathfrak{p}} = \begin{cases} 0 & \text{additive reduction} \\ 1 & \text{split multiplicative reduction} \\ -1 & \text{non-split multiplicative reduction} \end{cases}$$

In the case when E/K is an elliptic curve, and $A/\mathbb{Q}$ is the restriction of scalars of E to $\mathbb{Q}$, the abelian variety A has dimension equal to $[K : \mathbb{Q}]$. For every prime $p \in \mathbb{Q}$ of good reduction, the reduction A_p is isomorphic to the product of the Weil restrictions of $E_\mathfrak{p}$ to $\mathbb{F}_p$ as $\mathfrak{p}$ ranges over primes in $\mathcal{O}_K$ above p. Thus, we can compute $L(A, s)$ via the formula

$$L_{A,p}(T) = \prod_{\mathfrak{p}|p} L_{E,\mathfrak{p}}\left(T^{f_\mathfrak{p}}\right)$$

where $f_\mathfrak{p}$ is the residue degree of the prime $\mathfrak{p}$ above p in $\mathcal{O}_K$.

Our goal is to compute $L_{A,p}(T)$ for all but a predetermined finite set of primes $p < N$ in order to numerically approximate the L-function $L(A, s)$. For our project we used $N = 2^{30}$ as our target for a reasonable running time.

All computations are performed using SageMath [9] and Pari/GP [8].

§1.1.1 Applications

Much of the current numerical data we have on L-functions comes from the Jacobians of algebraic curves. Therefore, the L-functions of Weil restrictions of elliptic curves over number fields provide valuable new data. This has several applications, including the following:

1. Birch and Swinnerton-Dyer conjecture: It is conjectured that for abelian varieties A/K over a number field with L-function $L(A, s)$, there exists an analytic continuation of $L(A, s)$ to the entire complex plane, and its order of vanishing at 1 is equal to the rank of A as a finitely generated abelian group.

   For this purpose, computing the L-polynomial of the reductions up to $O(\sqrt{n})$, where n is the conductor of E, is sufficient to bound the rank. Although it is impossible to ascertain whether the ith derivative of $L(A, s)$ at s is equal to exactly 0, we can prove that it is nonzero by numerically approximating $L(A, s)$ and showing that 0 does not lie within the margin of error.

2. Sato-Tate Distributions: Associated to every abelian variety A over a number field K is a compact Lie group G, called the Sato-Tate group of A. The Sato-Tate Group is always a compact subgroup of the unitary symplectic group $\mathrm{USp}(2g)$, where g is the dimension of A. The unitary group consists of complex matrices P satisfying $P^*P = I$, and the symplectic group consists of complex matrices which preserve the skew-symmetric form $S = \left(\begin{smallmatrix} 0 & I \\ -I & 0 \end{smallmatrix}\right)$ (that is, $P^TSP = S$) [6, 9.1].

   Associated to locally every compact topological group G is a nonnegative, real-valued, countably additive ($\mu(\sum_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} \mu(A_i)$ for disjoint $A_i \subset G$), left-translation-invariant ($\mu(gS) = \mu(S)$ for $S \subset G$), nontrivial measure known as the Haar measure. The Haar measure always exists and is unique up to a positive scalar. If G is compact, $\mu(G)$ is positive and finite, so we can uniquely specify a normalized Haar measure by declaring $\mu(G) = 1$.

   Let A be an abelian variety over $\mathbb{Q}$ of dimension g. For every prime p we can take the coefficients $a_1, a_2, \ldots, a_g$ of the L-polynomial $\sum_{i=0}^{2g} a_i T^i$ and normalize them by dividing by $p^{i/2}$. We may then examine the distribution of these normalized coefficients as $p \to \infty$. A generalization of the Sato-Tate conjecture states that the distribution of the normalized a_i matches the distribution of characteristic polynomials in G according to the Haar measure.

   By computing the L-polynomials of A, we are able to visually see these distributions of normalized coefficients by plotting them on a histogram. From these histograms we can also infer information about the Sato-Tate group. For example, the number of 'distinct parts' in the histogram corresponds to the number of connected components in G as a topological space.

   For this application, we only care about the asymptotic behavior of the distribution of $a_i/p^{i/2}$, so we can completely ignore the bad primes.

§1.2 Factoring primes over $\mathcal{O}_K$

Let $f(x)$ be a minimal polynomial of K with root α, and let $\mathcal{O}$ denote the order $\mathbb{Z}[\alpha]$. Let $E/K : y^2 = h(x)$ be an elliptic curve, h a monic cubic. Declare p exceptional if p divides $\operatorname{disc}(f)$, $N(\operatorname{disc}(h))$, or $N(h(0))$. We have

$$\operatorname{disc}(f) = \operatorname{disc}(\mathcal{O}) = [\mathcal{O}_K : \mathcal{O}]^2 \operatorname{disc}(K)$$

[3, 12.4]. The condition $p \nmid \operatorname{disc}(f)$ implies the following:

- p is unramified, since $p \nmid \operatorname{disc}(K)$
- (p) is prime to the conductor of $\mathcal{O}$, since $p \nmid \frac{\operatorname{disc}(f)}{\operatorname{disc}(K)} = [\mathcal{O}_K : \mathcal{O}]^2$

Thus, the Dedekind-Kummer Theorem applies to all non-exceptional primes p [3, 6.33], and (p) factors in the maximal order $\mathcal{O}_K$ as

$$(p) = \prod_i (p, \overline{g}_i(\alpha))$$

where $\overline{g}_i$ is the lift of g_i in the factorization

$$f(x) = \prod_i g_i(x) \in \mathbb{F}_p[x]$$

Since p does not divide the discriminant of f, all of the g_i are distinct. If we let $\mathfrak{p}_i := (p, \overline{g}_i(\alpha))$, then the residue field $\mathcal{O}_K/\mathfrak{p}_i$ is isomorphic to $\mathbb{F}_p[x]/(g_i)$, with the reduction map sending α to a root of g_i in $\mathbb{F}_p$.

§1.3 Determining the traces mod p using the Hasse Invariant

The Hasse invariant $H_{p^r}(E)$ of an elliptic curve [4, Problem 4.1]

$$E/\mathbb{F}_q : y^2 = h(x)$$

over a finite field of characteristic p is defined as the coefficient of x^{p^r-1} in $h(x)^{\frac{p^r-1}{2}} \in \mathbb{F}_q[x]$, where h is a monic cubic. It can be shown that

$$\#E(\mathbb{F}_q) = 1 - H_q(E)$$

over $\mathbb{F}_q$, and hence over $\mathbb{F}_p$; it follows that for $\mathfrak{p}$ of good reduction, $H_{N(\mathfrak{p})}(E) = a_{\mathfrak{p}}$ holds as an identity over $\mathbb{F}_p$. We also have the recurrence relation

$$H_{p^{r+1}}(E) = H_{p^r}(E)H_p(E)^{p^r}$$

which yields the explicit formula

$$H_{p^r}(E) = H_p(E)^{\frac{p^r-1}{p-1}}$$

Hence, for any integer prime $p \in \mathbb{Z}$, we can determine $a_{\mathfrak{p}} \pmod p$ for all $\mathfrak{p} \mid p$ by computing the coefficient of x^{p-1} in $h(x)^{\frac{p-1}{2}}$ reduced modulo $(p) \subset \mathcal{O}_K$.

§1.3.1 Hasse-Witt Matrices

We now consider an elliptic curve $E/K : y^2 = h(x)$, with the coefficients of h lying in $\mathcal{O} = \mathbb{Z}[\alpha]$, where α is a root of the defining polynomial f of K. For prime $p \in \mathbb{Z}$, we will use $H_p(E)$ to denote the coefficient of x^{p-1} in $h(x)^{\frac{p-1}{2}}$, possibly mod (p).

We now want an algorithm to systematically determine the coefficient of $x^p - 1$ in $h(x)^{\frac{p-1}{2}}$ for all $p < N$. Our method is based on the one described in [5, Section 3].

Let h_m^n denote the coefficient of m in $h(x)^n$, and let v_m^n denote the row vector

$$v_m^n := [h_{m-d+1}^n, h_{m-d+2}^n, \dots, h_m^n] \in \mathbb{Z}[\alpha]^d$$

Examining the kth coefficient of $h(x)^{n+1} = h(x)h(x)^n$ implies that

$$h_m^{n+1} = \sum_{i=0}^{d} h_i h_{m-i}^n$$

and the $(k-1)$th coefficient of $\left[h(x)^{n+1}\right]' = (n+1)h'(x)h(x)^n$ implies that

$$mh_m^{n+1} = (n+1)\sum_{i=0}^{d} ih_i h_{m-i}^n$$

Subtracting the two yields

$$mh_0 h_m^n = \sum_{i=1}^{d} ((n+1)i - m)h_i h_{m-i}^n$$

This expresses h_m^n as a linear combination of $h_{m-1}^n, h_{m-2}^n, \dots, h_{m-d}^n$, which determines $d \times d$ integer matrices M_m^n in terms of $m, n, h_0, \dots, h_d$ which satisfy

$$mh_0 v_m^n = v_{m-1}^n M_m^n$$

Iteratively applying this identity yields

$$v_m^n = \frac{1}{mh_0} v_{m-1}^n M_m^n = \cdots = \frac{1}{m!h_0^m} v_0^n M_1^n \dots M_m^n$$

Since $v_0^n = [0, \dots, 0, h_0^n]$ and h_0^k is just h_0 raised to the kth power, we have

$$v_m^n = \frac{1}{h_0^{m-n} m!} V_0 M_1^n \dots M_m^n$$

where $V_0 = [0, \dots, 0, 1] \in \mathbb{Z}^d$.

In the case $d = 3$, M_m^n is given by

$$M_m^n := \begin{pmatrix} 0 & 0 & (3n+3-m)h_3 \\ mh_0 & 0 & (2n+2-m)h_2 \\ 0 & mh_0 & (n+1-m)h_1 \end{pmatrix}$$

If $p \nmid h_0$ is an odd prime and $n = \frac{p-1}{2}$, then modulo $(p) \subset \mathcal{O}_K$ we have the identity

$$\begin{aligned} v_{2n}^n &= \frac{1}{h_0^n (2n)!} V_0 M_1^n \dots M_m^n \\ &\overset{\text{Wilson}}{=} -\frac{1}{h_0^n} V_0 \prod_{m=1}^{2n} \frac{1}{2} \begin{pmatrix} & & (6n+6-2m)h_3 \\ 2mh_0 & & (4n+4-2m)h_2 \\ & 2mh_0 & (2n+2-2m)h_1 \end{pmatrix} \\ &= -\frac{1}{2^{2n} h_0^n} V_0 \prod_{m=1}^{2n} \begin{pmatrix} & & (3p+3-2m)h_3 \\ 2mh_0 & & (2p+2-2m)h_2 \\ & 2mh_0 & (p+1-2m)h_1 \end{pmatrix} \\ &\overset{\text{Fermat}}{=} -\frac{1}{h_0^n} V_0 \prod_{m=1}^{2n} \begin{pmatrix} & & (3-2m)h_3 \\ 2mh_0 & & (2-2m)h_2 \\ & 2mh_0 & (1-2m)h_1 \end{pmatrix} \end{aligned}$$

Note that Wilson's Theorem and Fermat's Little Theorem apply because $(2n)!$ and 2^{2n} are integers in $\mathbb{Z}$.

Thus, if we set

$$M_m := \begin{pmatrix} & & (3-2m)h_3 \\ 2mh_0 & & (2-2m)h_2 \\ & 2mh_0 & (1-2m)h_1 \end{pmatrix}$$

then we can compute $H_p(E) \equiv h_{2n}^n \mod (p)$ by computing the partial products of M_m up to $p - 1$, which only depend on m and the coefficients of h. We can then reduce $H_p(E)$ modulo each prime $\mathfrak{p} \mid p$, and obtain $a_\mathfrak{p}$ using properties of the Hasse Invariant as described above.

§1.3.2 Remainder Forest

We use the Remainder Forest algorithm implemented by Harvey and Sutherland in [5, Section 4.1] to compute the partial products $M_1 \dots M_{2n} \pmod p$ efficiently in both time and space.

The Remainder Forest algorithm only accommodates matrices in $\mathbb{Z}$. Here we contribute two new algorithms to extend Remainder Forest to orders $\mathbb{Z}[\alpha]$ in a general number field of arbitrary degree.

- One method is to express a matrix $R \in \mathbb{Z}[\alpha]^{3\times 3}$ as

$$R = R_0 + \alpha R_1 + \cdots + \alpha^{g-1} R_{g-1}$$

for $M_i \in \mathbb{Z}^{3\times 3}$. This allows us to compute

$$RS = \big(R_0 + \alpha R_1 + \cdots + \alpha^{g-1} R_{g-1}\big)\big(S_0 + \alpha S_1 + \cdots + \alpha^{g-1} S_{g-1}\big)$$

using g^2 matrix multiplications to obtain an expression

$$RS = T_0 + \alpha T_1 + \cdots + \alpha^{2g-2} T_{2g-2}$$

and by precomputing the vector α^k for $g \le k \le 2g-2$ in terms of the basis $1, \ldots, \alpha^{g-1}$, we need g^2 more scalar-matrix multiplications to express

$$RS = U_0 + \alpha U_1 + \cdots + \alpha^{g-1} U_{g-1}$$

Note that the matrix entries grow with n, while the scalars for converting α^k to the basis $1, \ldots, \alpha^{g-1}$ are constant in size. Thus, we can compute the product of two matrices in $\mathbb{Z}[\alpha]$ in time dominated by the g^2 matrix multiplications.

- Alternatively, instead of expressing $R \in \mathbb{Z}[\alpha]^{3\times 3}$ as a tuple of g integer matrices, we can modify R using the companion matrix of f. If $f = x^g + c_{g-1}x^{g-1} + \cdots + c_0$ is a monic polynomial, its companion matrix C_f is given by

$$C_f := \begin{pmatrix} 0 & 0 & \ldots & 0 & -c_0 \\ 1 & 0 & \ldots & 0 & -c_1 \\ 0 & 1 & \ldots & 0 & -c_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & 1 & -c_{g-1} \end{pmatrix}$$

It can be shown that, if $p = a_0 + a_1 x + \cdots + a_{g-1}x^{g-1}$ and $q = b_0 + b_1 x + \cdots + b_{g-1}x^{g-1}$ in $\mathbb{Z}[x]/(f)$ and $pq = d_0 + d_1 x + \cdots + d_{g-1}x^{g-1}$, then

$$\left(\sum_{i=0}^{g-1} a_i C_f^i\right)\left(\sum_{i=0}^{g-1} b_i C_f^i\right) = \sum_{i=0}^{g-1} d_i C_f^i$$

The vector $[d_0, \ldots, d_{g-1}]$ is equal to the leftmost column of $\sum_{i=0}^{g-1} d_i C_f^i$.

This suggests running remainder forest after replacing every entry of R with the transformation

$$a_0 + a_1\alpha + \cdots + a_{g-1}\alpha^{g-1} \mapsto a_0 I + a_1 C_f + \cdots + a_{g-1} C_f^{g-1}$$

and taking the leftmost column of each $g \times g$ submatrix at the end. This method takes the same number of matrix multiplications, but the matrix dimension is now $3g \times 3g$.

We perform $r \times r$ matrix multiplication by transforming the r^2 entries, multiplying the matrices of Fourier coefficients, and performing r^2 inverse transforms. We have $r = O(g)$, and if we take g to be fixed and n to vary, then asymptotically we expect the running time of matrix multiplication to be dominated by the $2r^2$ Fourier transforms. Therefore, in both cases we expect the number of multiplications to increase by a factor of g^2.

The only other change in the time complexity is the bit size of the matrix entries. Since the leftmost column of the (i,j)th $g \times g$ submatrix in the second method is identical to the (i,j)th entries of $R_0, R_1, \ldots, R_{g-1}$ in the first method, we expect the bit complexity of each matrix entry to be the same between both methods.

Therefore, we expect the asymptotic time complexity of both modifications to Remainder Forest to be the same. However, note the second modification uses $3g \times 3g$ matrices in lieu of g 3×3 matrices. We checked for various elliptic curves that each of the $g \times g$ submatrices contains around g times as many bits as the corresponding entries of the g 3×3 submatrices in total. Thus, the second modification takes approximately g times more memory.

We opted for the second modification in spite of the memory consumption. This is because the implementation of Remainder Forest in C by Harvey and Sutherland only works for integer matrices, so in order to implement the first method, one would have to modify the Remainder Forest library's code to accommodate multiplication of tuples of matrices.

The parameter κ

The remainder forest algorithm uses a parameter κ, where we separately compute 2^κ subtrees instead of computing a single remainder tree. Modifying κ creates a tradeoff between time and space. We chose to optimize for time, and found that $\kappa = 7$ was the optimal value for N in the range of interest. Increasing κ to 8 would take more time but use around half as much memory.

§1.4 Finding the actual traces using BSGS

We use a modified verison of the Baby Steps Giant Steps group algorithm (BSGS) [4, 8.9] to compute the actual value of $a_{\mathfrak{p}} \in \mathbb{Z}$ given its value mod p. Let $E/\mathbb{F}_q$ be an elliptic curve, $q = p^d$. Its cardinality $\#E(\mathbb{F}_q)$ lies in the interval Hasse Interval

$$\mathcal{H}_q = [(\sqrt{q} - 1)^2, (\sqrt{q} + 1)^2]$$

BSGS first chooses a random point $P \in E$, and then tries to determine a multiple of the order of P by computing a series of

- Baby steps: $0, -P, \dots, -(r-1)P$
- Giant steps: $aP, (a+r)P, \dots, (a+(s-1)r)P$

where a is the bottom of the Hasse interval, and r, s are chosen so that $rs > 4\sqrt{q}$. Since every number in the Hasse interval can be expressed as the difference of one giant and one baby step, we are guaranteed to find at least one pair (bP, gP) of a baby step and giant step such that $bP = gP$. The complexity is given by the square root of the size of the interval in which we search; in this case, $O(r + s) = O(q^{1/4})$ elliptic curve operations. If we choose $r = s = 2q^{1/4}$, then the number of operations is $4q^{1/4}$ asymptotically.

There are many optimizations to the constant factor [4, 8.11]. The one of most relevance to us is 'fast inverses,' in which we compute $-P_{\text{baby}}$ at the same time as P_{baby} by negating the y-coordinate. This allows us to collect $2r$ baby steps at the cost of computing r. The condition on r and s is loosened to $2rs > 4\sqrt{q}$, so for the optimal running time we may reduce r and s by a factor of $\sqrt{2}$.

In the case of our algorithm, we also know the value of $a_q \pmod{p}$, so we only need to search an arithmetic progression with $4\sqrt{q}/p$ elements. This gives rise to the following generalized algorithm to find $\#E(\mathbb{F}_q)$, given its value modulo some arbitrary modulus m:

Algorithm 1.4.1

We compute the following series of baby steps and giant steps:

- Baby steps: $0, \pm mP, \dots, \pm rmP$
- Giant steps: $aP, (a + rm)P, \dots, (a + srm)P$

where a is chosen near the bottom of the Hasse interval such that $a \equiv 1 - a_q \pmod{m}$, and r, s are chosen so that $2rs = \frac{4\sqrt{q}}{m}$.

We remark that BSGS is usually implemented using a hash table that terminates at the first collision it encounters. However, during our implementation of BSGS in SageMath, we found that Python's dictionary type takes a disproportionate amount of time hashing tuples of finite field elements, so we instead opted to use a list which we sort at the end.

We also remark that all operations are conducted in affine coordinates, as is the default in Pari/GP and SageMath. There is also the option of batching field inversions [4, 8.10] when converting from projective to affine coordinates, thereby reducing the asymptotic complexity by a factor of $\log n$ (where $n = \log q$). However, doing so would require writing our own implementation of elliptic curve operations on SageMath or Pari/GP, which we opted not to do for this project.

§1.4.1 Mestre's Theorem

Let $\lambda(G)$ denote the group exponent of a group G, that is, the LCM of the orders of all the elements of G. When $E/\mathbb{F}_q$ is an elliptic curve over a finite field, its abelian group is given by $\mathbb{Z}/n\mathbb{Z} \oplus \mathbb{Z}/N\mathbb{Z}$, with $n \mid N$; then $\lambda(E(\mathbb{F}_q)) = N$. Let $E'/\mathbb{F}_q$ be the quadratic twist of E. We can relate their traces by

$$\#E(\mathbb{F}_q) + \#E'(\mathbb{F}_q) = 2q + 2$$

Mestre's Theorem [7] states that if $q > 229$, then either $\lambda(E(\mathbb{F}_q))$ or $\lambda(E'(\mathbb{F}_q))$ has a unique multiple in $\mathcal{H}_q$. This suggests implementing BSGS as a Las Vegas algorithm in which we choose points $P \in E$ and $P' \in E'$ one after the other, and update an LCM of point orders in $E(\mathbb{F}_q)$ and $E'(\mathbb{F}_q)$ until either one exceeds $4\sqrt{q}$, in which case the divisor's unique lift in $\mathcal{H}_q$ is the desired cardinality of either $E(\mathbb{F}_q)$ or $E'(\mathbb{F}_q)$.

§1.4.2 Computing the trace mod 2 and 3

To further reduce the constant factor, we attempt to compute the trace mod 2 and 3.

Trace mod 2

First, 2 divides $\#E(\mathbb{F}_q)$ if and only if E has points of 2-torsion of the form $(x_0, 0)$. This occurs if and only if $h(x)$ has roots in $\mathbb{F}_q$. Thus, we can determine $a_p \mod 2$ by finding roots of h; this has expected complexity of $O(n\mathbf{M}(n))$.

This allows us to improve the constant factor of BSGS by $\sqrt{2}$.

Trace mod 3

To compute the trace of Frobenius t mod 3, we have the following strategy:

1. Factor the 3-division polynomial ψ_3, which is a degree 4 polynomial whose roots over $\overline{\mathbb{F}_q}$ are the distinct x-coordinates of the 3-torsion subgroup $E[3]$. Let g be an irreducible factor of ψ_3 with maximal degree m.

2. Let π_3 denote the restriction of the Frobenius endomorphism $(x : y : z) \mapsto (x^q : y^q : z^q)$ to $E[3]$.

   If $m = 2$, let P, Q be the points in $E[3]$ whose x-coordinates are the distinct roots of g. Note that distinct x-coordinates in $E[3]$ implies linear independence, since all nonzero points in the subspace $\langle P \rangle = \{0, P, -P\}$ have the same x-coordinate as P. Thus, $\{P, Q\}$ is a basis of $E[3]$, and π_3 sends P to $\pm Q$ and vice versa.

   Thus, the Frobenius element is conjugate to $\begin{pmatrix} & * \\ * & \end{pmatrix}$, hence $t \equiv 0 \pmod 3$.

3. If $m = 1$ or $m = 3$, then there exists a subspace $V = \{0, P, -P\}$ whose x-coordinate x_0 lies in $\mathbb{F}_q$. This subspace is fixed by π_3.

$m = 1$: If $m = 1$, then π_3 fixes all subspaces of $E[3]$. Thus, it must be a scalar matrix.

If $f(x_0)$ is a square, then in particular the y-coordinates of P are also fixed by π_3. Thus, π_3 is given by $\begin{pmatrix} 1 & \\ & 1 \end{pmatrix}$, so we conclude $t \equiv 2 \pmod 3$.

If $f(x_0)$ is nonsquare, then π_3 sends P to $-P$, so π_3 is given by $\begin{pmatrix} 2 & \\ & 2 \end{pmatrix}$. We conclude $t \equiv 1 \pmod 3$.

$m = 3$: If $m = 3$, we claim that π_3 cannot be diagonalizable over $\overline{\mathbb{F}_3}$. Assume the contrary; then since π_3 fixes V, it must be diagonalizable over $\mathbb{F}_3$. But then π_3 fixes another subspace W different from V, which means that π_3 must fix the x-coordinate of W as well. This yields another root of ψ_3 in $\mathbb{F}_q$, contradicting $m = 3$.

Thus, π_3 must be given by $\left(\begin{smallmatrix} a & b \\ & a \end{smallmatrix}\right)$. It acts on V via multiplication by a.

If $f(x_0)$ is a square, then π_3 fixes the y-coordinates of P as well, so $a = 1$ and thus $t \equiv 2 \pmod 3$.

If $f(x_0)$ is nonsquare, then π_3 swaps $\pm P$, so $a = 2$ and thus $t \equiv 1 \pmod 3$.

In either case, $t \equiv 2 \pmod 3$ if $f(x_0)$ is square, and $t \equiv 1 \pmod 3$ otherwise.

We remark that in all of the above subcases, $\det \pi_3 = 1$ in $\mathbb{F}_3$. Thus, we necessarily have $q \equiv 1 \pmod 3$.

4. Otherwise, $m = 4$. No subspace of $E[3]$ is fixed, so π_3 is neither diagonalizable over $\mathbb{F}_3$ nor defective. Hence π_3 must be diagonalizable over $\mathbb{F}_9$, so its order must divide $3^2 - 1 = 8$. Since π_3 is not diagonalizable over $\mathbb{F}_3$, its order must be 4 or 8.

   If the eigenvalues of π_3 are $a \pm b\sqrt{2} \in \mathbb{F}_9 \setminus \mathbb{F}_3$ (b nonzero), then note that $\operatorname{tr} \pi_3 = 0$ if and only if $a = 0$ if and only if the eigenvalues are ± 2 if and only the order of π_3 is 4.

   We claim that $\operatorname{tr} \pi_3 \neq 0$. Assume otherwise; examining the conjugacy classes in $\mathrm{GL}_2(\mathbb{F}_3)$ with trace 0 implies that π_3 is conjugate to $\left(\begin{smallmatrix} & * \\ * & \end{smallmatrix}\right)$. But this implies that π_3^2 is a scalar matrix, so π_3 has order ≤ 2 as a permutation of the x-coordinates, which contradicts the irreducibility of ψ_3.

   Thus, we conclude that the $t \not\equiv 0 \pmod 3$. Also note that eigenvalues of π_3 are $a \pm b\sqrt{2}$ for $a, b \neq 0$, so
$$q \equiv \det \pi_3 \equiv a^2 + b^2 \equiv 2 \pmod 3$$

In conclusion, we have the following:

- If $m = 2$, then $a_q \equiv 0 \pmod 3$
- If $m = 1$ or 3, then $a_q \equiv 1 \pmod 3$ if $f(x_0)$ is nonsquare, and $a_q \equiv 2 \pmod 3$ if $f(x_0)$ is square. This occurs only if $q \equiv 1 \pmod 3$.
- If $m = 4$, then $a_q \not\equiv 0 \pmod 3$. This occurs only if $q \equiv 2 \pmod 3$.

The running time of this computation is dominated by factoring ψ_3, which has complexity $O(n\mathbf{M}(n))$. For $m \leq 3$ we obtain the exact value of $a_q \pmod 3$, which improves the constant factor by $\sqrt{3}$. For $m = 4$ we at least know that $a_q \not\equiv 0 \pmod 3$, which allows us to skip 1/3 of the baby steps, improving the constant factor by $\sqrt{3/2}$.

Trace mod $\ell > 3$

We also considered using Schoof's algorithm for $\ell \geq 3$, which performs addition and multiplication in the endomorphism ring $\operatorname{End}(E[\ell]) \simeq \mathbb{F}_q[x]/(\psi_\ell(x))$ and searches for c such that $\pi_\ell^2 - c\pi_\ell + q = 0$. However, we found that operations in relative ring extensions in SageMath were too costly and outweighed the $\sqrt{\ell}$ decrease in the constant term for p in the range of interest.

Torsion Information

We can compute the torsion structure of E/K at the very beginning. If the torsion subgroup G is nontrivial, then G injects into the abelian group $E_\mathfrak{p}$ for all $\mathfrak{p} \nmid |G|$ [10, Prop VII.3.1]. Thus, we know $a_\mathfrak{p} \pmod{|G|}$ for almost all p, which we can incorporate to further save a constant factor in the BSGS (or save the time of computing $a_\mathfrak{p} \pmod{2, 3}$).

Although 100% of elliptic curves over a number field have trivial torsion, a sizeable number in the LMFDB have nontrivial torsion, so this optimization will see some use.

§1.4.3 Separate Implementations for $d = 1, 2$ and Complexity Considerations

When $d = 1$, the value of $a_{\mathfrak{p}} \pmod p$ uniquely determines $a_{\mathfrak{p}}$ for all $p : p > 4\sqrt{p}$, that is, for all $p \geq 17$. Thus, we can determine $a_{\mathfrak{p}}$ given its value $\pmod p$ in essentially $O(1)$ time.

When $d = 2$, the Hasse Interval contains $4p + 1$ integers. In this case we always have $q \equiv 1 \pmod 3$, so we can always determine $a_{\mathfrak{p}} \pmod{2, 3}$ using the steps outlined above. This pinpoints the value of $a_{\mathfrak{p}} \pmod{6p}$, so we can again find its unique lift in the Hasse interval. The complexity is $O((\log p)\mathbf{M}(\log p))$ to factor the relevant polynomials.

When $d \geq 3$, the complexity is dominated by the $O(p^{d/4-1/2})$ elliptic curve operations. In our implementation, each operation has bit complexity $O(\mathbf{I}(d \log p)) = O(\mathbf{M}(d \log p) \log \log p)$.

We primarily used this algorithm for cubic number fields, in which case $d = 3$ is the maximum residue field degree. For the primes $p \leq 2^{30}$ of interest, the total cost of $O(p^{1/4})$ operations over all primes, although asymptotically exponential, is still overshadowed by the overall cost of remainder forest. However, for number fields of higher degree $g \geq 4$, the $O(p^{1/2})$ operations at every prime become dominant. Modifications would need to be made to remainder forest to compute the values of $H_p(E) \pmod{p^r}$ for some appropriate value of r; see [11, 4.4].

§1.5 Summary of algorithm

1. Let E/K be an elliptic curve over a number field K. Let $f(x)$ be a minimal polynomial of K with root α. Find a short Weierstrass model $y^2 = h(x)$ for the elliptic curve E with coefficients in the order $\mathcal{O} = \mathbb{Z}[\alpha]$. Declare p exceptional if p divides $\operatorname{disc}(f)$, $N(\operatorname{disc}(h))$, or $N(f(0))$.

2. Call a modified version of remainder forest using the companion matrix of f to determine the Hasse Invariant $H_p(E) \in \mathcal{O}_K/(p)$ for all non-exceptional primes p. For every prime $\mathfrak{p} \subset \mathcal{O}_K$ above p, further reduce $H_p(E)$ in $\mathcal{O}_K/\mathfrak{p}$ and compute
$$a_{\mathfrak{p}} = H_p(E)^{\frac{N(\mathfrak{p})-1}{p-1}} \in \mathbb{F}_p \subset \mathcal{O}_k/\mathfrak{p}$$

3. Use the Baby-Steps Giant-Steps group algorithm on the reduction $E_{\mathfrak{p}}$ to compute the actual value of $a_{\mathfrak{p}}$ given its value mod p.

4. Use the formula $L_{A,p}(T) = \prod_{\mathfrak{p}|p} L_{E,\mathfrak{p}}(T^{f_{\mathfrak{p}}})$ to compute the L-polynomial of the restriction of scalars of E to $\mathbb{Q}$.

§1.6 Timings

We ran the entire algorithm on MIT's computer Babbage and recorded the timings (in hours) for $N = 2^{22}, 2^{24}, 2^{26}, 2^{28}, 2^{30}$. We used number field $K = \mathbb{Q}[\alpha]$, where $\alpha^3 - \alpha^2 + \alpha - 2 = 0$, and the Weil restriction of the elliptic curve $E_1/K : y^2 = x^3 - x + \alpha$. We used $\kappa = 7$ for remainder forest. The primes we excluded were $\{2, 3, 83, 131\}$.

lg N	RForest	BSGS
22	0.21	0.13
24	0.98	0.72
26	3.72	1.83
28	17.50	3.57
30	90.20	15.18

§1.7 Final Data

For the same elliptic curve $E_1/K : y^2 = x^3 - x + \alpha$ over $\mathbb{Q}[x]/(x^3 - x^2 + x - 2)$, we created histograms of the computed distributions of the normalized a_1, a_2, a_3 of the Weil restriction using Sutherland's computer program. The histogram also includes the moments computed from our data. The reader may wish to compare the following diagrams with the expected Sato-Tate group of E_1, see [12].

The full animated histograms, as well as the final code used for this project, may be found at https://github.com/danielhu-mit/l-poly-weil-restriction.

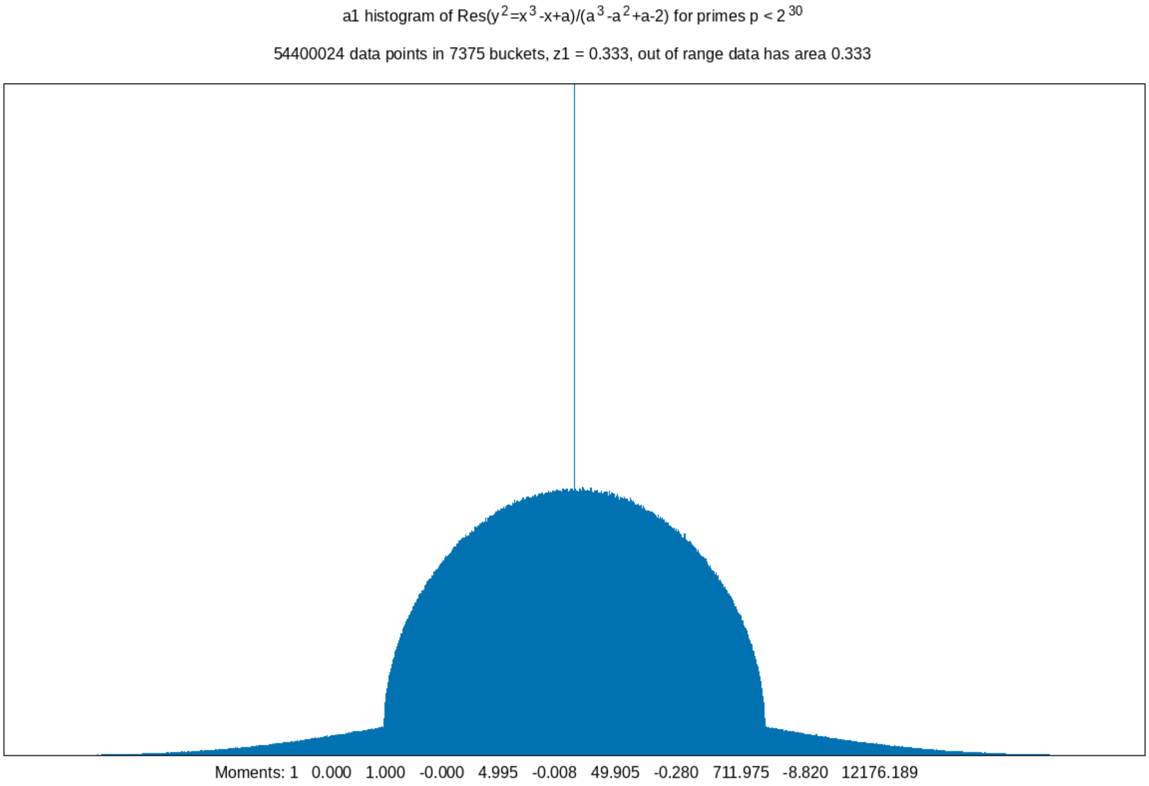

Figure 1.1: Distribution of a_1

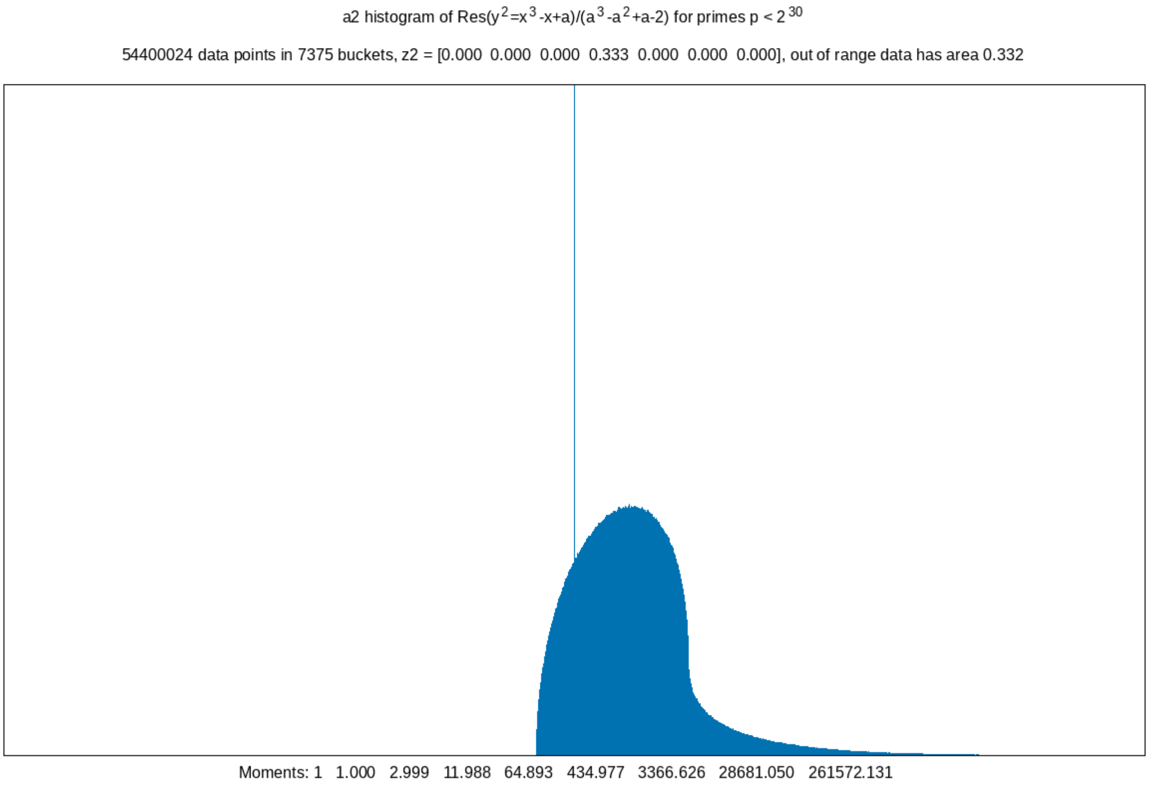

Figure 1.2: Distribution of a_2

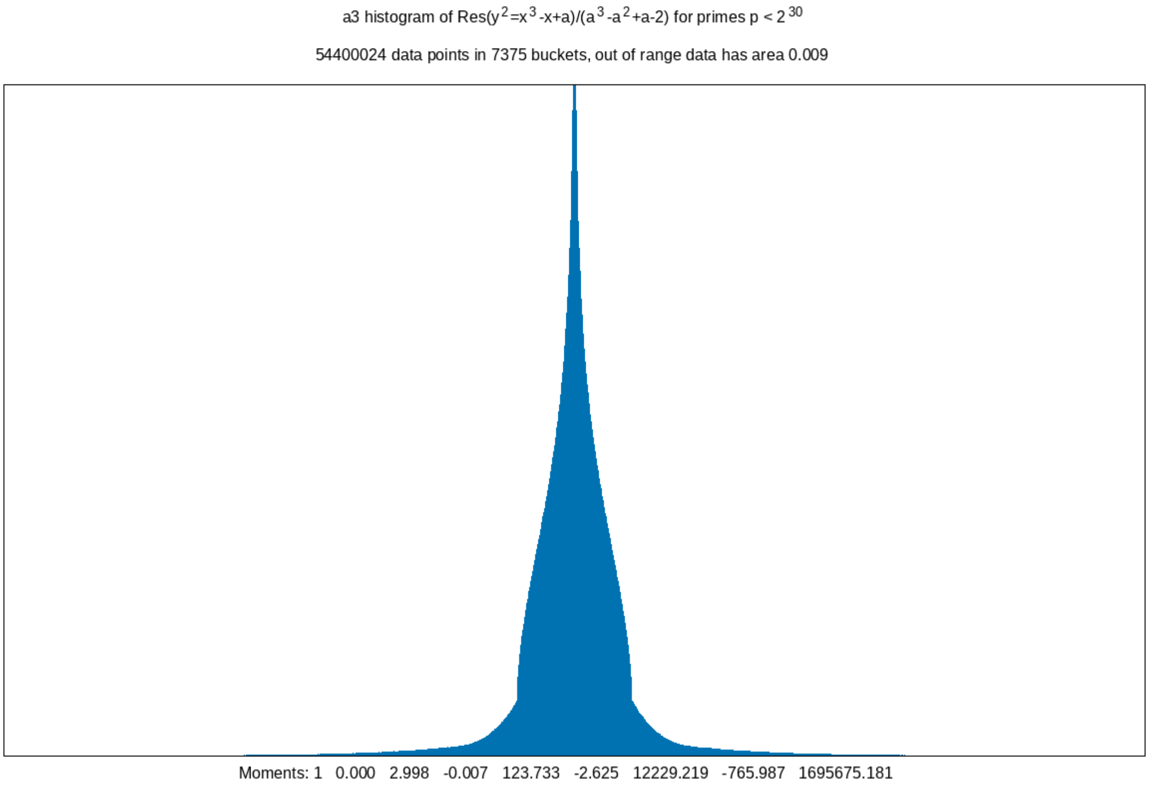

Figure 1.3: Distribution of a_3

Bibliography

[1] André Weil, Numbers of solutions of equations in finite fields, Bulletin of the American Mathematical Society 55 (1949), 497–508.

[2] Bjorn Poonen, *Rational Points on Varieties*, American Mathematical Society, 2017, pp. 145, 212

[3] Andrew V. Sutherland, *18.785 Fall 2019 Lecture Notes*, Sept 2019.

[4] Andrew V. Sutherland, *18.783 Spring 2021 Lecture Notes*, Feb 2021.

[5] David Harvey and Andrew V. Sutherland, *Computing Hasse-Witt matrices of hyperelliptic curves in average polynomial time*, LMS Journal of Computation and Mathematics 17 (2014), 257-273, https://arxiv.org/abs/1402.3246v1.

[6] M. Artin, *Algebra*, 2nd ed., Pearson Education, 2011.

[7] John E. Cremona and Andrew V. Sutherland, *On a theorem of Mestre and Schoof*, Journal de Théorie des Nombres de Bordeaux 22 (2010), 353-358, https://arxiv.org/abs/0901.0120.

[8] The PARI-Group, 'PARI/GP version Version 2.11.2', Bordeaux (2018), available at http://pari.math.u-bordeaux.fr/.

[9] The Sage Developers, 'SageMath, the Sage Mathematics Software System (Version 9.1)' (2020), available at http://www.sagemath.org/.

[10] Joseph H. Silverman, *The Arithmetic of Elliptic Curves*, 2nd ed., Springer, 2009.

[11] David Harvey, *Computing zeta functions of arithmetic schemes*, 2014, https://arxiv.org/pdf/1402.3439.

[12] The L-functions and modular forms database, *Sato-Tate group $E_{s,t}$ of weight 1 and degree 6*, https://www.lmfdb.org/SatoTateGroup/1.6.E.6.1a.