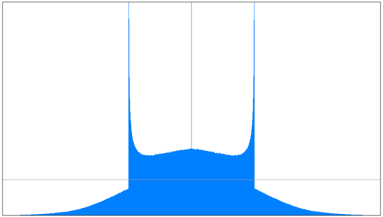
Computing Hasse-Witt matrices of hyperelliptic curves in average polynomial time

David Harvey and Andrew Sutherland

ANTS XI — August 9, 2014



Motivation

Let $C/\mathbb{Q}$ be a smooth projective curve of genus g.

For each prime p of good reduction we have the trace of Frobenius

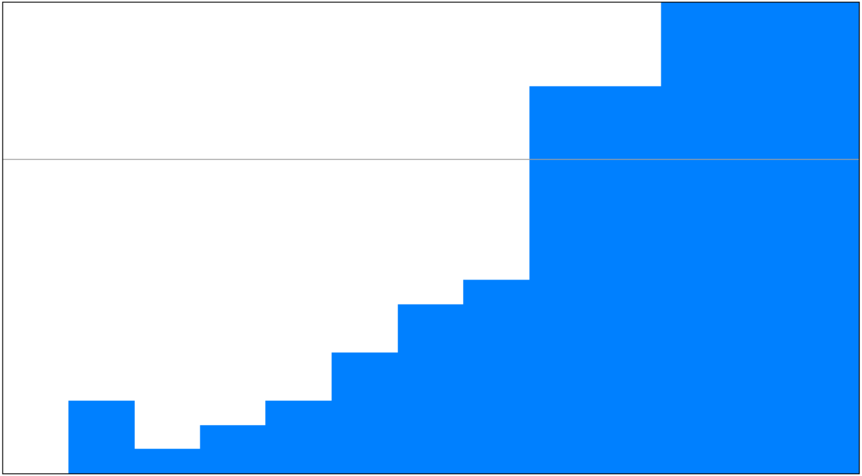
$$t_p = p + 1 - N_p \in [-2g\sqrt{p}, 2g\sqrt{p}],$$

where $N_p = \#C(\mathbb{F}_p)$, and the normalized trace

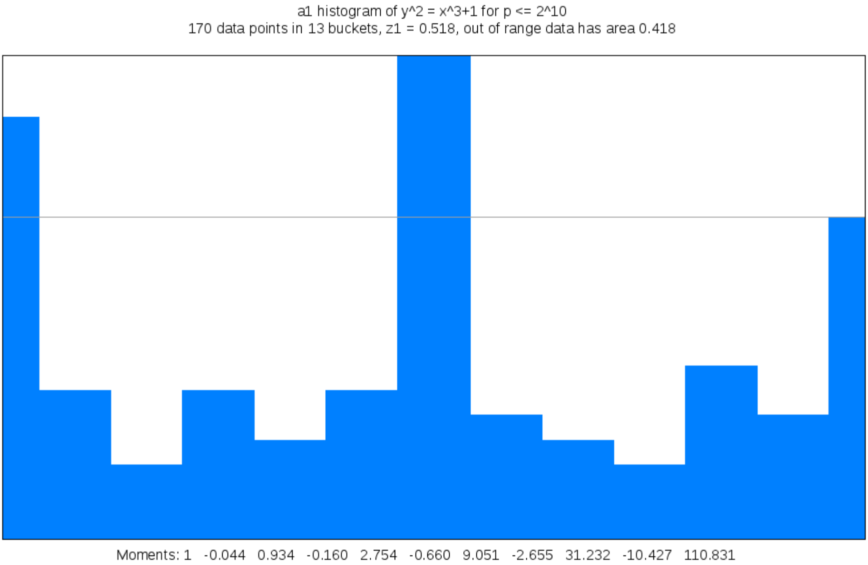
$$x_p = t_p/\sqrt{p} \in [-2g, 2g].$$

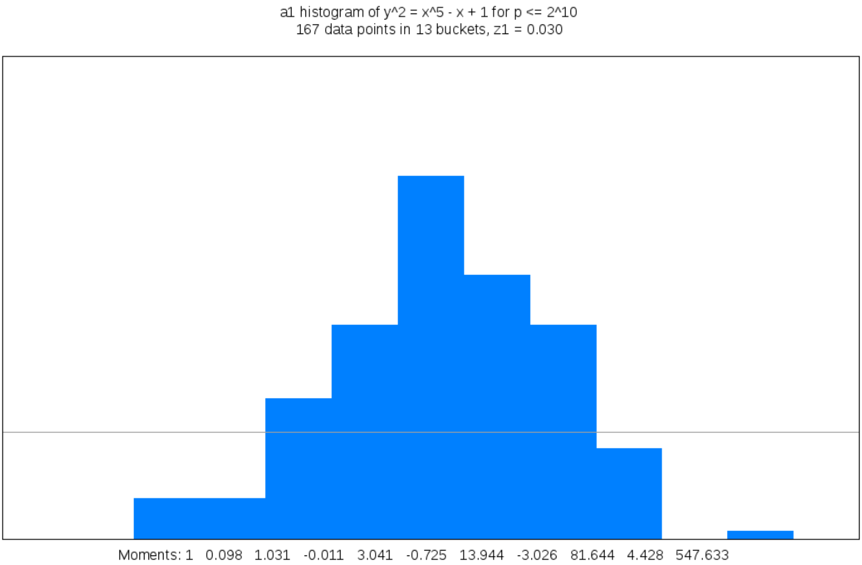
What is the distribution of x_p?

a1 histogram of y^2 + xy + y = x^3 - x^2 - 20067762415575526585033208209338542750930230312178956502x
+ 34481611795030556467032985690390720374855944359319180361266008296291939448732243429 for p <= 2^10
164 data points in 13 buckets, out of range data has area 0.262

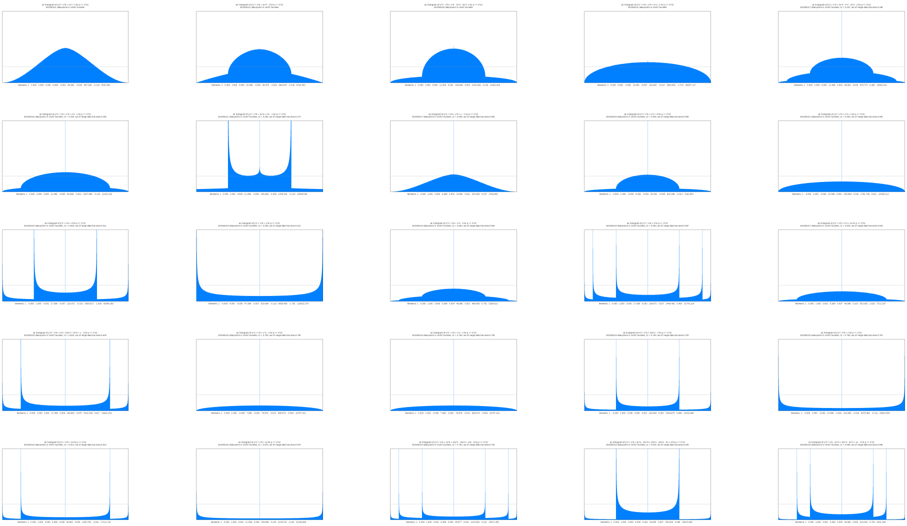


Moments: 1 1.087 1.797 2.656 4.662 7.555 13.659 23.304 42.847 75.617 140.495


a1 histogram of y^2 = x^3+1 for p <= 2^10
170 data points in 13 buckets, z1 = 0.518, out of range data has area 0.418
Moments: 1 -0.044 0.934 -0.160 2.754 -0.660 9.051 -2.655 31.232 -10.427 110.831


a1 histogram of y^2 = x^5 - x + 1 for p <= 2^10
167 data points in 13 buckets, z1 = 0.030
Moments: 1 0.098 1.031 -0.011 3.041 -0.725 13.944 -3.026 81.644 4.428 547.633

Exceptional trace distributions of genus 2 curves $C/\mathbb{Q}$



L-polynomial distributions

For a smooth projective curve $C/\mathbb{Q}$ of genus g and a prime p of good reduction for C we have the *zeta function*

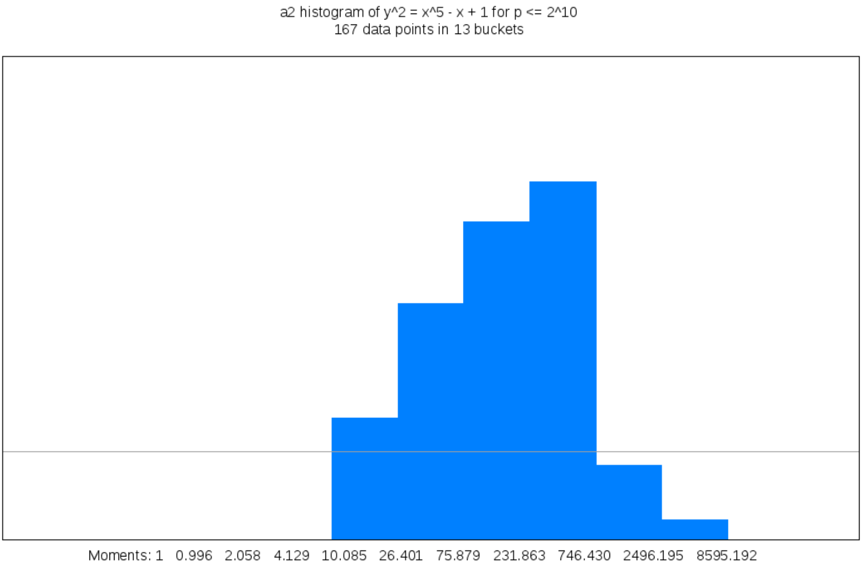
$$Z_p(T) := \exp\left(\sum_{k=1}^{\infty} N_k T^k/k\right) = \frac{L_p(T)}{(1-T)(1-pT)},$$

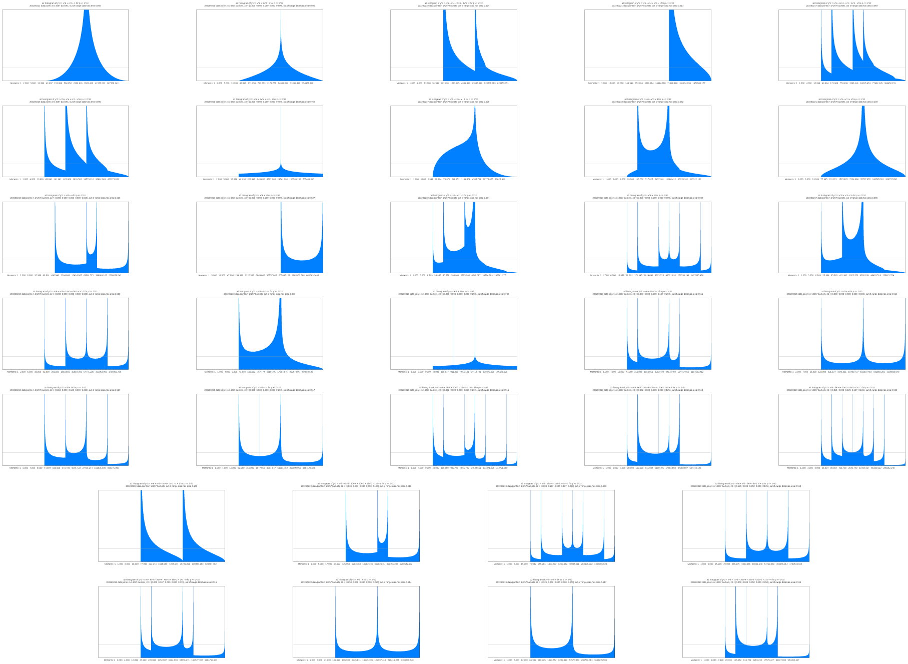
where $L_p \in \mathbb{Z}[T]$ has degree $2g$. The normalized L-polynomial

$$\bar{L}_p(T) := L_p(T/\sqrt{p}) = \sum_{i=0}^{2g} a_i T^i \in \mathbb{R}[T]$$

is monic, reciprocal ($a_i = a_{2g-i}$), and unitary (roots on the unit circle). The coefficients a_i satisfy the Weil bounds $|a_i| \leq \binom{2g}{i}$.

We may now consider the distribution of $a_1, a_2, \ldots, a_g$ as p varies.


a2 histogram of y^2 = x^5 - x + 1 for p <= 2^10
167 data points in 13 buckets
Moments: 1 0.996 2.058 4.129 10.085 26.401 75.879 231.863 746.430 2496.195 8595.192



Computing zeta functions

Algorithms to compute $L_p(T)$ for low genus hyperelliptic curves

	complexity (ignoring factors of $O(\log\log p)$)		
algorithm	$g = 1$	$g = 2$	$g = 3$
point enumeration	$p \log p$	$p^2 \log p$	$p^3 \log p$
group computation	$p^{1/4} \log p$	$p^{3/4} \log p$	$p^{5/4} \log p$
p-adic cohomology	$p^{1/2} \log^2 p$	$p^{1/2} \log^2 p$	$p^{1/2} \log^2 p$
CRT (Schoof-Pila)	$\log^5 p$	$\log^8 p$	$\log^{12} p$

Computing zeta functions

Algorithms to compute $L_p(T)$ for low genus hyperelliptic curves

algorithm	complexity (ignoring factors of $O(\log\log p)$) $g=1$	$g=2$	$g=3$
point enumeration	$p\log p$	$p^2\log p$	$p^3\log p$
group computation	$p^{1/4}\log p$	$p^{3/4}\log p$	$p^{5/4}\log p$
p-adic cohomology	$p^{1/2}\log^2 p$	$p^{1/2}\log^2 p$	$p^{1/2}\log^2 p$
CRT (Schoof-Pila)	$\log^5 p$	$\log^8 p$	$\log^{12} p$

(see [Kedlaya-S, ANTS VIII]).

An average polynomial-time algorithm

All of these methods perform separate computations for each p.
But we want to compute $L_p(T)$ for all good $p \le N$ using reductions of *the same curve* in each case. Can we take advantage of this?

An average polynomial-time algorithm

All of these methods perform separate computations for each p.
But we want to compute $L_p(T)$ for all good $p \leq N$ using reductions of *the same curve* in each case. Can we take advantage of this?

Theorem (H 2012)

There exists a deterministic algorithm that, given a hyperelliptic curve $y^2 = f(x)$ of genus g with a rational Weierstrass point and an integer N, computes $L_p(T)$ for all good primes $p \leq N$ in time

$$O\big(g^{8+\epsilon} N \log^{3+\epsilon} N\big),$$

assuming the coefficients of $f \in \mathbb{Z}[x]$ have size bounded by $O(\log N)$.

Average time is $O\big(g^{8+\epsilon} \log^{4+\epsilon} N\big)$ per prime, polynomial in g and $\log p$.

An average polynomial-time algorithm

	complexity (ignoring factors of $O(\log\log p)$)		
algorithm	$g=1$	$g=2$	$g=3$
point enumeration	$p\log p$	$p^2\log p$	$p^3\log p$
group computation	$p^{1/4}\log p$	$p^{3/4}\log p$	$p^{5/4}\log p$
p-adic cohomology	$p^{1/2}\log^2 p$	$p^{1/2}\log^2 p$	$p^{1/2}\log^2 p$
CRT (Schoof-Pila)	$\log^5 p$	$\log^8 p$	$\log^{12} p$
Average polytime	$\log^4 p$	$\log^4 p$	$\log^4 p$

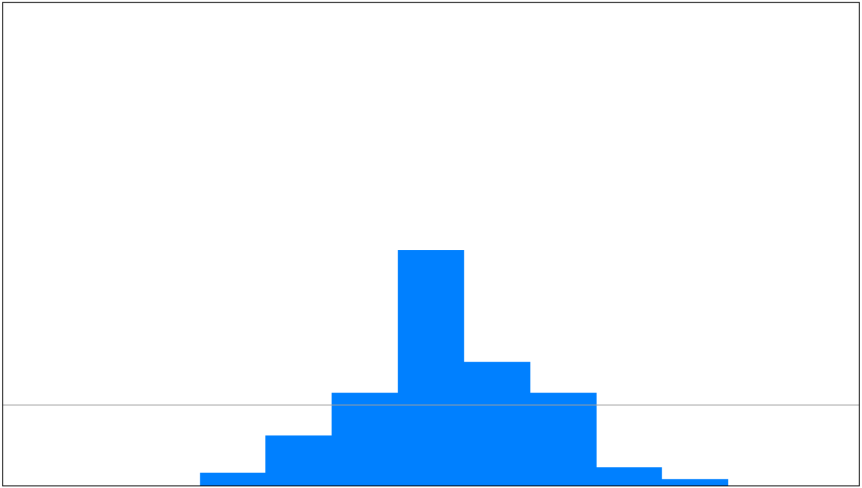
But is it practical?

	genus 2			genus 3		
N	**smalljac**	paper	current	**hypellfrob**	paper	current
2^{14}	0.2	0.4	0.1	6.8	2.0	0.3
2^{15}	0.6	1.1	0.3	15.6	5.5	1.0
2^{16}	1.7	2.8	0.8	37.6	13.6	2.7
2^{17}	5.6	6.8	1.8	95.0	33.3	7.0
2^{18}	20.2	16.8	4.7	250	80.4	16.3
2^{19}	76.4	39.7	11.1	681	192	38.7
2^{20}	257	94.4	26.0	1920	459	91.7
2^{21}	828	227	61.4	5460	1090	212
2^{22}	2630	534	142	16300	2540	489
2^{23}	8570	1240	321	49400	5940	1120
2^{24}	28000	2920	729	152000	13800	2540
2^{25}	92300	6740	1660	467000	31800	6510
2^{26}	316000	15800	3800	1490000	72900	16600

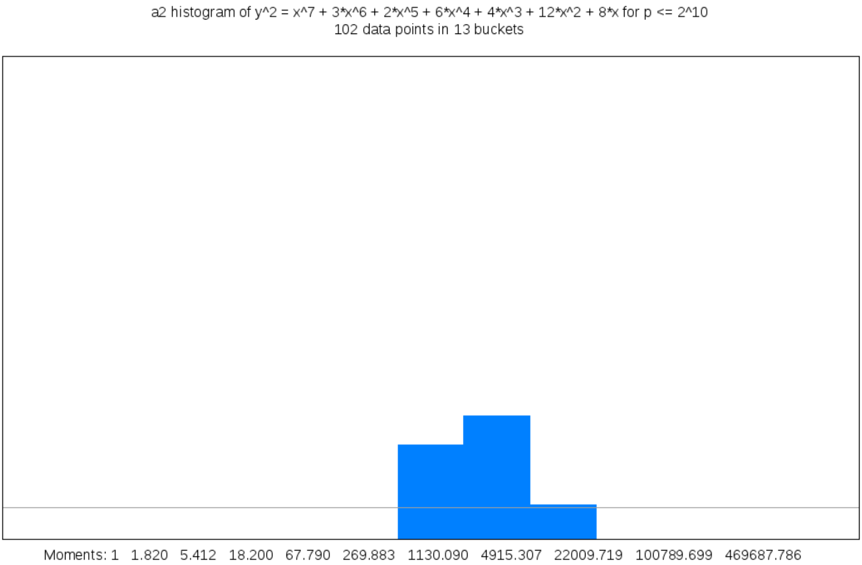
Comparison of average polynomial time algorithm (as in the paper and currently) to **smalljac** in genus 2 and **hypellfrob** in genus 3.

(Intel Xeon E5-2670 2.6 GHz CPU seconds).

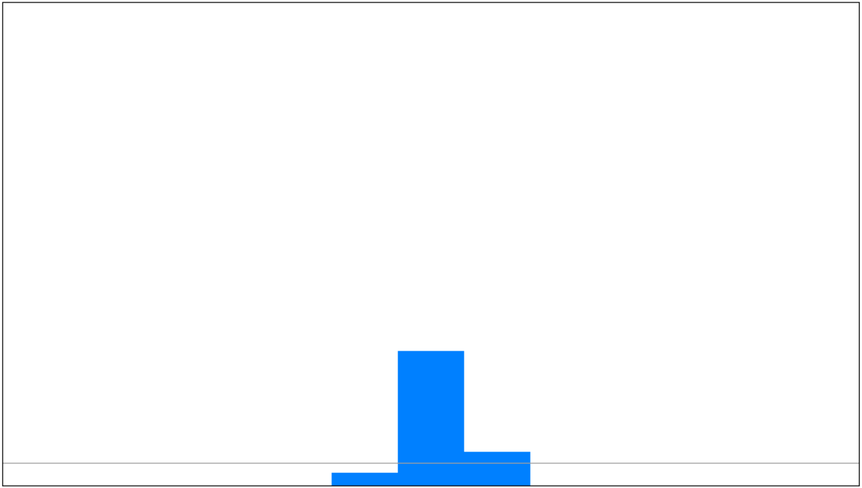
a1 histogram of y^2 = x^7 + 3*x^6 + 2*x^5 + 6*x^4 + 4*x^3 + 12*x^2 + 8*x for p <= 2^10
102 data points in 13 buckets, z1 = 0.255



Moments: 1 0.240 1.553 1.151 6.748 7.896 41.260 66.465 312.792 632.344 2726.539


a2 histogram of y^2 = x^7 + 3*x^6 + 2*x^5 + 6*x^4 + 4*x^3 + 12*x^2 + 8*x for p <= 2^10
102 data points in 13 buckets
Moments: 1 1.820 5.412 18.200 67.790 269.883 1130.090 4915.307 22009.719 100789.699 469687.786

a3 histogram of y^2 = x^7 + 3*x^6 + 2*x^5 + 6*x^4 + 4*x^3 + 12*x^2 + 8*x for p <= 2^10
102 data points in 13 buckets



Moments: 1 0.562 4.394 8.788 76.249 206.830 1833.874 5645.427 50662.626 165983.954 1505626.828

The algorithm in genus 1

The Hasse invariant h_p of an elliptic curve $y^2 = f(x) = x^3 + ax + b$ over $\mathbb{F}_p$ is the coefficient of x^{p-1} in the polynomial $f(x)^{(p-1)/2}$.

We have $h_p \equiv t_p \bmod p$, which uniquely determines t_p for $p > 13$.

Naïve approach: iteratively compute $f, f^2, f^3, \ldots, f^{(N-1)/2}$ in $\mathbb{Z}[x]$ and reduce the x^{p-1} coefficient of $f(x)^{(p-1)/2}$ mod p for each prime $p \leq N$.

The algorithm in genus 1

The Hasse invariant h_p of an elliptic curve $y^2 = f(x) = x^3 + ax + b$ over $\mathbb{F}_p$ is the coefficient of x^{p-1} in the polynomial $f(x)^{(p-1)/2}$.

We have $h_p \equiv t_p \bmod p$, which uniquely determines t_p for $p > 13$.

Naïve approach: iteratively compute $f, f^2, f^3, \ldots, f^{(N-1)/2}$ in $\mathbb{Z}[x]$ and reduce the x^{p-1} coefficient of $f(x)^{(p-1)/2}$ mod p for each prime $p \leq N$.

But the polynomials f^n are huge, each has $\Omega(n^2)$ bits.
It would take $\Omega(N^3)$ time to compute $f, \ldots, f^{(N-1)/2}$ in $\mathbb{Z}[x]$.

So this is a terrible idea...

The algorithm in genus 1

The Hasse invariant h_p of an elliptic curve $y^2 = f(x) = x^3 + ax + b$ over $\mathbb{F}_p$ is the coefficient of x^{p-1} in the polynomial $f(x)^{(p-1)/2}$.

We have $h_p \equiv t_p \bmod p$, which uniquely determines t_p for $p > 13$.

Naïve approach: iteratively compute $f, f^2, f^3, \ldots, f^{(N-1)/2}$ in $\mathbb{Z}[x]$ and reduce the x^{p-1} coefficient of $f(x)^{(p-1)/2}$ mod p for each prime $p \leq N$.

But the polynomials f^n are huge, each has $\Omega(n^2)$ bits.
It would take $\Omega(N^3)$ time to compute $f, \ldots, f^{(N-1)/2}$ in $\mathbb{Z}[x]$.

So this is a terrible idea...

But we don't need all the coefficients of f^n, we only need one, and we only need to know its value modulo $p = 2n + 1$.

A better approach

Let $f(x) = x^3 + ax + b$, and let f_k^n denote the coefficient of x^k in $f(x)^n$. Using $f^n = f \cdot f^{n-1}$ and $(f^n)' = nf'f^{n-1}$, one obtains linear relations

$$
\begin{aligned}
(n+2)f_{2n-2}^n &= n\left(2af_{2n-3}^{n-1} + 3bf_{2n-2}^{n-1}\right),\\
(2n-1)f_{2n-1}^n &= n\left(3f_{2n-4}^{n-1} + af_{2n-2}^{n-1}\right),\\
2(2n-1)bf_{2n}^n &= (n+1)af_{2n-4}^{n-1} + 3(2n-1)bf_{2n-3}^{n-1} - (n-1)a^2f_{2n-2}^{n-1}.
\end{aligned}
$$

These allow us to compute the vector $v_n = [f_{2n-2}^n, f_{2n-1}^n, f_{2n}^n]$ from the vector $v_{n-1} = [f_{2n-4}^{n-1}, f_{2n-3}^{n-1}, f_{2n-2}^{n-1}]$ via multiplication by a 3×3 matrix M_n:

$$v_n = v_0 M_1 M_2 \cdots M_n.$$

For $n = (p-1)/2$, the Hasse invariant of the elliptic curve $y^2 = f(x)$ over $\mathbb{F}_p$ is obtained by reducing the third entry f_n^{2n} of v_n modulo p.

Computing $t_p \bmod p$

To compute $t_p \bmod p$ for all odd primes $p \leq N$ it suffices to compute

$$\begin{aligned} M_1 &\bmod 3 \\ M_1 M_2 &\bmod 5 \\ M_1 M_2 M_3 &\bmod 7 \\ &\vdots \\ M_1 M_2 M_3 \cdots M_{(N-1)/2} &\bmod N \end{aligned}$$

Doing this naïvely would take $O(N^{2+\epsilon})$ time.

But it can be done in $O(N^{1+\epsilon})$ time using a *remainder tree*.
For best results, use a *remainder forest*.

The algorithm in genus g.

The *Hasse-Witt* matrix of a hyperelliptic curve $y^2 = f(x)$ over $\mathbb{F}_p$ of genus g is the $g \times g$ matrix $W_p = [w_{ij}]$ with entries

$$w_{ij} = f_{pi-j}^{(p-1)/2} \bmod p \qquad (1 \leq i, j \leq g).$$

The w_{ij} can each be computed using recurrence relations between the coefficients of f^n and those of f^{n-1}, as in genus 1.

The algorithm in genus g.

The *Hasse-Witt* matrix of a hyperelliptic curve $y^2 = f(x)$ over $\mathbb{F}_p$ of genus g is the $g \times g$ matrix $W_p = [w_{ij}]$ with entries

$$w_{ij} = f_{pi-j}^{(p-1)/2} \bmod p \qquad (1 \le i, j \le g).$$

The w_{ij} can each be computed using recurrence relations between the coefficients of f^n and those of f^{n-1}, as in genus 1.

The congruence

$$L_P(T) \equiv \det(I - TW_p) \bmod p$$

allows us to determine the coefficients $a_1, \ldots, a_g$ of $L_p(T)$ modulo p.

The algorithm can be extended to compute $L_p(T)$ modulo higher powers of p (and thereby obtain $L_p \in \mathbb{Z}[T]$), but for $g \le 3$ it is faster in practice to derive $L_p(T)$ from $L_p(T) \bmod p$ using computations in $\mathrm{Jac}(C)$.

Complexity

Theorem (HS 2014)

Given a hyperelliptic curve $y^2 = f(x)$ of genus g, and an integer N, one can compute the Hasse-Witt matrices W_p for all good primes $p \le N$ in

$$O\big(g^{2+\epsilon}\, N \log^{3+\epsilon} N\big) \text{ time} \qquad \text{and} \qquad O(g^2 N) \text{ space,}$$

provided that g and $\log \|f\|$ are sufficiently small relative to N.

The time bound has improved by a factor of $g^{3-\epsilon}$ since the paper.
The complexity is quasi-linear in the output size.

This should extend to computing $L_p \in \mathbb{Z}[T]$ in $O(g^{4+\epsilon} N \log^{3+\epsilon} N)$ time.

In progress: generalize to non-hyperelliptic curves.