Efficient Point-Counting Algorithms for Superelliptic Curves

Matthew Hase-Liu

Abstract

In this paper, we present efficient algorithms for computing the number of points and the order of the Jacobian group of a superelliptic curve over finite fields of prime order p. Our method employs the Hasse-Weil bounds in conjunction with the Hasse-Witt matrix for superelliptic curves, whose entries we express in terms of multinomial coefficients. We present a fast algorithm for counting points on specific trinomial superelliptic curves and a slower, more general method for all superelliptic curves. For the first case, we reduce the problem of simplifying the entries of the Hasse-Witt matrix modulo p to a problem of solving quadratic Diophantine equations. For the second case, we extend Bostan et al.'s method for hyperelliptic curves to general superelliptic curves. We believe the methods we describe are asymptotically the most efficient known point-counting algorithms for certain families of trinomial superelliptic curves.

1 Introduction

In this paper, we present and prove asymptotics for the fastest known algorithms for counting the number of points on certain families of possibly singular plane curves. A central problem in number theory is the study of rational solutions of polynomial equations. Even before the development of algebra, the Greeks were interested in systematically determining all rational solutions to the Pythagorean equation: $a^2+b^2=c^2$. More recently, Andrew Wiles proved Fermat's Last Theorem, which states that Fermat's equation, $a^n+b^n=c^n$, has no nontrivial rational solutions — a problem that had withstood over 350 years of effort by mathematicians.

To study points on a curve defined by a polynomial equation, it is often helpful to keep track of the field where they "live." Solutions to Fermat's equation are called $\mathbb{Q}$-rational because we are concerned with solutions over the field $\mathbb{Q}$. More generally, solutions (x, y) to polynomial equations $f(x, y) = 0$ are called K-rational points, if both x and y are in a field K. In particular, mathematicians are concerned with $\mathbb{F}_{p^r}$-rational points (where p is some prime) because they 1) are easier to study than $\mathbb{Q}$-rational points, 2) have analogues to complex analysis, and 3) are useful in investigating and understanding $\mathbb{Q}$-rational points [15]. In this paper, we focus on counting $\mathbb{F}_p$-points on the curve defined by a particular type of irreducible polynomial equation:

$$0 = y^a - x^b f(x) = y^a - \left(m_c x^{b+c} + \ldots + m_0 x^b\right).$$

The curve C defined by $y^a - x^b f(x) = 0$ is commonly known as a superelliptic curve.

Advances in the field of modern algebraic geometry have brought forth a wealth of techniques to further the study of $\mathbb{F}_{p^r}$-rational points on plane curves.[1] We define $\#C\left(\mathbb{F}_{p^r}\right)$ to be the number of $\mathbb{F}_{p^r}$-rational points on a smooth projective curve C and $\#J_C\left(\mathbb{F}_p\right)$ to be the number of points in the Jacobian of C, an abelian group canonically associated with the curve. In the early 1940s, André Weil proved the Riemann hypothesis for curves over finite fields, which reveals a great deal

[1] A plane curve is the loci of a single bivariate polynomial equation over a field.

about the relationship between $\#C(\mathbb{F}_{p^r})$ and a curve's zeta function. A direct consequence is the Hasse-Weil bounds, which state that

$$|p^r + 1 - \#C(\mathbb{F}_{p^r})| \leq 2g\sqrt{p^r} \text{ and } \left(\sqrt{p^r} - 1\right)^{2g} \leq \#J_C(\mathbb{F}_{p^r}) \leq \left(\sqrt{p^r} + 1\right)^{2g}, \tag{1}$$

where g is the genus of C [8].

In this paper, we describe fast algorithms for computing $\#C(\mathbb{F}_p)$ and $\#J_C(\mathbb{F}_p)$ for trinomial (when $y^a - x^b f(x)$ consists of three monomials) and general superelliptic curves, even if they are singular at the origin. To do so, we employ a matrix associated with C that encodes information about $\#C(\mathbb{F}_p)$ and $\#J_C(\mathbb{F}_p)$ modulo p (called the Hasse-Witt matrix) and explicitly compute its entries in terms of multinomial coefficients. We show that these multinomial coefficients are actually binomial for trinomial superelliptic curves, which allows us to reduce the problem of simplifying the Hasse-Witt matrix modulo p to a problem of solving Diophantine equations of the form $x^2 + dy^2 = m$. Assuming $p > 16g^2$ and $\gcd(ac, p-1) = 3, 4, 6$ or 8, we use both this simplified Hasse-Witt matrix and the Hasse-Weil bounds to compute $\#C(\mathbb{F}_p)$ probabilistically in $O(\mathrm{M}(\log p)\log p)$ time, deterministically in $O\left(\mathrm{M}(\log p)\log^2 p \log\log p\right)$ (assuming the generalized Riemann hypothesis), and deterministically in $O\left(\mathrm{M}(\log^3 p)\frac{\log^2 p}{\log\log p}\right)$ time, where $\mathrm{M}(n)$ is the time needed to multiply two n-digit numbers. Similarly, we can compute $\#J_C(\mathbb{F}_p)$ with the same running times for $g = 2$ and with running time $O\left(\sqrt{p}\right)$ if $g = 3$, assuming that a also divides $p - 1$. For general superelliptic curves, we extend the work of Bostan et al. [2] to simplify the Hasse-Witt matrix modulo p in $O\left(\mathrm{M}\left(\sqrt{p}\log p\right)\right)$ time. Again, assuming $p > 16g^2$ (and additionally $g = 2$ or 3 for $\#J_C(\mathbb{F}_p)$), we use this simplified Hasse-Witt matrix and the Hasse-Weil bounds to also compute $\#C(\mathbb{F}_p)$ and $\#J_C(\mathbb{F}_p)$ in $O\left(\mathrm{M}\left(\sqrt{p}\log p\right)\right)$ time.

Our work has salient applications in mathematics, cryptography, and coding theory. The Hasse-Witt matrix can be used to find the L-polynomial of a curve, which is intimately connected to the zeta function of a curve — the central object of study in the Weil-conjectures. In addition, Fité et

al.'s classification of Sato-Tate distributions in curves of genus two required extensive back-and-forth between theoretical work and computations of $\#C(\mathbb{F}_p)$ [6]. Fast point-counting algorithms will be essential in studying the Sato-Tate groups of curves of higher genus. Moreover, $J_C(\mathbb{F}_p)$ is a finite abelian group, so a public-key cryptosystem can be constructed with it. Using the Jacobian of a curve of higher genus (typically 3 or 4) allows us to achieve a comparable level of security using a smaller field size, even after taking into account known attacks on the discrete logarithm problem for curves of genus $g > 2$. For example, hyperelliptic curve cryptosystems can have field sizes half of those used in elliptic curve cryptosystems [8]. To build such cryptosystems, computing $\#J_C(\mathbb{F}_p)$ is essential to ascertain which Jacobians are cryptographically secure in practice.[2]

The organization of this paper is as follows. In Section 2, we provide background information about regular differentials on a curve, the Cartier operator, and the Hasse-Witt matrix. In Section 3, we use the Cartier operator to compute the entries of the Hasse-Witt matrix in terms of multinomial coefficients. In Section 4, we give fast point-counting algorithms and their time complexities for both trinomial and general superelliptic curves. We conclude in Section 5 with a brief discussion on future work.

2 Background

In this section, we recall several facts about regular differentials on a curve, the Cartier operator, the Hasse-Witt matrix, and $\#C(\mathbb{F}_p)$ and $\#J_C(\mathbb{F}_p)$.

2.1 Basic Facts

In this paper, we consider smooth projective plane curves C with affine model $y^a = x^b f(x) = x^b (m_c x^c + m_{c-1} x^{c-1} + \ldots + m_0)$ over a finite field $\mathbb{F}_p$, where $p \neq 2, 3$ is a prime, $y^a - x^b f(x)$ is irreducible, $f(x) \in \mathbb{F}_p[x]$, $a, c \geq 2, b \geq 0, m_c \neq 0, m_0 \neq 0$, and $p > a, b + c$.

[2] For a Jacobian to be secure, $\#J_C(\mathbb{F}_p)$ is required to be a product of a very small integer and a very large prime.

Let P be the convex hull of the set of lattice points in the interior of the Newton polygon of $g(x,y) = y^a - x^b f(x)$. If we take a side of P and label the lattice points that lie on it $0, 1, \ldots, s$, construct a "side polynomial" $h(t)$ with degree $s+1$ such that the coefficient of t^i is the coefficient of g corresponding to the lattice point i.

We impose the following two conditions on our superelliptic curves:

1. The curves $C/\mathbb{Z}$ and $C/\mathbb{F}_p$ have the same genus.

2. The "side polynomials" are square-free.

We denote the set of points on a curve C over the field $\mathbb{F}_p$ as $C(\mathbb{F}_p)$ and the Jacobian variety as $J_C(\mathbb{F}_p)$.

2.2 Regular Differentials and the Cartier Operator

Given a smooth projective variety C with coordinate ring $k[C]$, we can define $\Omega(C)$, the $k[C]$-module of regular differentials as elements of the form $\sum_i g_i df_i$ with $f_i, g_i \in k[C]$ subject to

- $d(f+g) = df + dg$;
- $d(fg) = fdg + gdf$;
- $da = 0$ if $a \in k$.

It is worth noting that $\Omega(C)$ is a g-dimensional vector space over the field k, where g is the genus of C.

Let ω be in $\Omega(C)$, where p is the prime corresponding to the finite field $\mathbb{F}_p$ over which C is defined. One can always find $a_0, a_1, \ldots, a_{p-1}$ such that $\omega = \left(a_0^p + a_1^p x + \ldots + a_{p-1}^p x^{p-1}\right) dx$ [3]. We can then define the Cartier operator as follows.

Definition 2.1. The Cartier operator $\mathscr{C}$ is a map from $\Omega(C)$ to itself such that

$$\mathscr{C}(\omega) = a_{p-1} dx.$$

The following are useful properties of the Cartier operator that we use later in writing down the Hasse-Witt matrix [9].

Lemma 2.2. *If F is the function field of C, then for all $\omega \in \Omega(C)$ and $f \in F/\mathbb{F}_p$,*

- $\mathscr{C}(f^p\omega) = f\mathscr{C}(\omega)$;
- $\mathscr{C}^n(x^j dx) = \begin{cases} 0, & \text{if } p^n \nmid j+1; \\ x^{(j+1)/p^n - 1}, & \text{otherwise.} \end{cases}$

2.3 Finding $\#C(\mathbb{F}_p)$ and $\#J_C(\mathbb{F}_p)$ with the Hasse-Witt Matrix

The Hasse-Witt matrix is the matrix associated with a smooth curve over some finite field and the action of the Frobenius endomorphism with respect to a basis of regular differentials for $\Omega(C)$.

Definition 2.3. Let $\omega_1, \ldots, \omega_g$ be a basis for $\Omega(C)$. Then, define the Hasse-Witt matrix $A_p(C)$ as $\left[a_{i,j}^{1/p}\right]$, where

$$\mathscr{C}(\omega_i) = \sum_{j=1}^{g} a_{i,j}\omega_j.$$

Since we are dealing with only finite fields with prime order, matrix $\left[a_{i,j}^{1/p}\right]$ is the same as matrix $[a_{i,j}]$ by Fermat's little theorem. The following theorem shows how this matrix encodes information about $\#C(\mathbb{F}_p)$ and $\#J_C(\mathbb{F}_p)$ [16].

Theorem 2.4. *Let $A_p(C)$ be the Hasse-Witt matrix of a smooth curve C and $\chi_p(t)$ be the characteristic polynomial of the Frobenius endomorphism. Then*

$$\chi_p(t) \equiv (-1)^g t^g \det(A_p(C) - tI) \pmod{p},$$

where I is the identity matrix.

It is well known that $\chi_p(t)$ is a polynomial of degree twice that of the genus g and that the coefficient of x^{2g-1} in $\chi_p(t)$ is simply the negative of the trace of $A_p(C)$ by the Cayley–Hamilton theorem. The trace of Frobenius, $\mathrm{tr}(A_p(C))$, satisfies

$$\mathrm{tr}(A_p(C)) = p + 1 - \#C(\mathbb{F}_p).$$

We can thus compute $\#C(\mathbb{F}_p)$ and $\#J_C(\mathbb{F}_p)$ modulo p by noting that

$$\#C(\mathbb{F}_p) \equiv 1 - \mathrm{tr}(A_p(C)) \pmod{p} \tag{2}$$

and

$$\#J_C(\mathbb{F}_p) = \chi_p(1) \equiv (-1)^g \det(A_p(C) - I) \pmod{p}. \tag{3}$$

If p is sufficiently large, we can then uniquely determine $\#C(\mathbb{F}_p)$ and narrow down the candidates for $\#J_C(\mathbb{F}_p)$ using the Hasse-Weil bounds.

3 Computing Hasse-Witt Matrices

In this section, we express the entries of the Hasse-Witt matrix of the superelliptic curve

$$C : y^a = x^b f(x) = x^b \left(m_c x^c + m_{c-1} x^{c-1} + \ldots + m_0\right) \tag{4}$$

over $\mathbb{F}_p$ explicitly as a sum of multinomial coefficients, where p is some prime bigger than a and $b + c$, $f(0) \neq 0$, and $\deg(f) = c$. It is well-known [5] that

$$\left\{ \omega_{i,j} = \frac{x^{i-1} y^{j-1}}{y^{a-1}} \middle| (i,j) \in \mathcal{N} \right\}$$

is a basis of regular differentials for $\Omega(C)$, where $\mathcal{N}$ is the set of lattice points in the interior of the Newton polygon of $y^a - x^b f(x)$:

$$\mathcal{N} = \left\{ (i,j) \middle| \begin{array}{c} 1 \le i \le b \text{ and } \left\lfloor -\frac{a}{b}i + a \right\rfloor + 1 \le j \le \left\lceil -\frac{a}{b+c}i + a \right\rceil - 1 \\ \text{or} \\ b+1 \le i \le b+c-1 \text{ and } 1 \le j \le \left\lceil -\frac{a}{b+c}i + a \right\rceil - 1 \end{array} \right\}.$$

The following lemmas will be used in our proof of Theorem 3.8.

Lemma 3.1. *If* $\gcd(p,a) = 1$ *and* $1 \le j \le a$*, there exist unique integers* u_j *and* v_j *such that* $j - 1 + (a-1)(p-1) = p(u_j - 1) + av_j$*, where* $1 \le u_j \le a$ *and* $0 \le v_j \le p-1$.

Proof. Consider the Diophantine equation $p(x-1) + ay = j - 1 + (a-1)(p-1)$. If we take this equation modulo p, we get $y \equiv a^{-1}(j - 1 + (p-1)(a-1)) \pmod{p}$. Let v_j be the unique integer such that $v_j \equiv y \pmod{p}$ and $0 \le v_j \le p-1$. Then, it remains to show that $u_j = \frac{j-1+(a-1)(p-1)-av_j}{p} + 1$ is between 1 and a, inclusive. Note that $av_j \equiv j-1+(a-1)(p-1) \pmod{p}$ and $0 \le av_j \le ap - a$, so $av_j \le j - 1 + (a-1)(p-1)$. Thus, $u_j \ge 1$. Also, since $j \le a$, we have $j - 1 + (a-1)(p-1) - av_j \le pa - p$, so $u_j \le a$. □

Remark 3.2. It turns out that the u_j are pairwise distinct, i.e. $u_j = u_{j'} \Rightarrow j = j'$. To see this, note that $u_j = u_{j'}$ implies that $a(v_j - v_{j'}) = j - j'$, so $a | j - j'$. However, $1 \le j, j' \le a-1$, so $0 \le j - j' \le a - 2$. Thus, $j = j'$.

Definition 3.3. Let $(i,j) \in \mathcal{N}$ and $k_0, k_1, \ldots$, and k_c be integers such that $k_0 + k_1 + \ldots + k_c = v_j, 0 \le k_0, \ldots, k_c \le v_j$, and $p | bv_j + ck_c + (c-1)k_{c-1} + \ldots + k_1 + i$. Then, define

$$s_{i,j}(k_0, \ldots, k_c) := \frac{bv_j + ck_c + (c-1)k_{c-1} + \ldots + k_1 + i}{p}.$$

Lemma 3.4. $u_j \le \left\lceil -\frac{a}{b+c} s_{i,j}(k_0, \ldots, k_c) + a \right\rceil - 1$.

Proof. Note that $\left\lfloor \frac{a}{b+c} s_{i,j}(k_0,\ldots,k_c)\right\rfloor + \frac{j}{p} = \left\lfloor \frac{a}{b+c}\frac{bv_j+ck_c+(c-1)k_{c-1}+\ldots+k_1+i}{p}\right\rfloor + \frac{j}{p} \leq \left\lfloor \frac{a((b+c)v_j+i)}{p(b+c)}\right\rfloor + \frac{a-\left\lfloor \frac{ai}{b+c}\right\rfloor -1}{p} \leq \frac{av_j+a+\left\{\frac{ai}{b+c}\right\}-1}{p}$, so $\left\lfloor \frac{a}{b+c} s_{i,j}(k_0,\ldots,k_c)\right\rfloor \leq \frac{av_j+a-j+\left\{\frac{ai}{b+c}\right\}-1}{p} < \frac{av_j+a-j}{p}$. However, $\left\lfloor \frac{a}{b+c} s_{i,j}(k_0,\ldots,k_c)\right\rfloor$ and $\frac{av_j+a-j}{p} = a-u_j$ are both integers, so $\left\lfloor \frac{a}{b+c} s_{i,j}(k_0,\ldots,k_c)\right\rfloor \leq \frac{av_j+a-j}{p}-1$. Using the fact that $-\lfloor x\rfloor = \lceil -x\rceil$, we find that $u_j = \frac{j-av_j-a}{p}+a \leq \left\lceil -\frac{a}{b+c} s_{i,j}(k_0,\ldots,k_c)+a\right\rceil -1$, as desired. □

Lemma 3.5. $\left\lfloor -\frac{a}{b} s_{i,j}(k_0,\ldots,k_c)+a\right\rfloor +1 \leq u_j$ *for* $1 \leq s_{i,j}(k_0,\ldots,k_c) \leq b$.

Proof. We consider two cases: $1 \leq i \leq b$ and $b+1 \leq i \leq b+c-1$. If $1 \leq i \leq b$, note that $\frac{av_j+a-j}{p} \leq \frac{av_j+\left\lceil \frac{ai}{b}\right\rceil -1}{p} < \frac{av_j+\frac{ai}{b}}{p} \leq \frac{a(bv_j+ck_c+(c-1)k_{c-1}+\ldots+k_1+i)}{pb} = \left\lceil \frac{a}{b} s_{i,j}(k_0,\ldots,k_c)\right\rceil$. Analogously, if $b+1 \leq i \leq b+c-1$, we have $\frac{av_j+a-j}{p} < \left\lceil \frac{a}{b} s_{i,j}(k_0,\ldots,k_c)\right\rceil$. In both situations, since $\frac{av_j+a-j}{p}$ and $\left\lceil \frac{a}{b} s_{i,j}(k_0,\ldots,k_c)\right\rceil$ are integers, we have $\left\lceil \frac{a}{b} s_{i,j}(k_0,\ldots,k_c)\right\rceil \geq \frac{av_j+a-j}{p}+1$. Thus $u_j = \frac{j-av_j-a}{p}+a \geq \left\lfloor -\frac{a}{b} s_{i,j}(k_0,\ldots,k_c)+a\right\rfloor +1$, as desired. □

Lemma 3.6. $1 \leq u_j$ *for* $c+1 \leq s_{i,j}(k_0,\ldots,k_c) \leq b+c-1$.

Proof. Note that $u_j = \frac{j-av_j-a}{p}+a \geq \frac{j-a(p-1)-a}{p}+a = \frac{j}{p} > 0$. We know u_j is an integer, so $u_j \geq 1$, as desired. □

Let $a_{(i,j),(i',j')}$ be the entry of Hasse-Witt matrix corresponding to coefficient of $\omega_{i',j'}$ when $\mathscr{C}(\omega_{i,j})$ is written as a linear combination of the basis elements. In particular, $a_{(i,j),(i,j)}$ would refer to a diagonal entry on the Hasse-Witt matrix. We adopt the following notation for convenience.

Definition 3.7. Define

$$\begin{bmatrix} a \\ b \end{bmatrix} := \begin{cases} \binom{a}{b}, & \text{if } a,b \in \mathbb{Z}_{\geq 0}; \\ 0, & \text{otherwise.} \end{cases}$$

and

$$\begin{bmatrix} a \\ b_1, b_2, \ldots, b_k \end{bmatrix} := \begin{cases} \binom{a}{b_1,b_2,\ldots,b_k}, & \text{if } a, b_i \in \mathbb{Z}_{\geq 0} \text{ and } \sum_i b_i = a; \\ 0, & \text{otherwise.} \end{cases}$$

Theorem 3.8. *The Hasse-Witt matrix of a superelliptic curve is given by* $\left[a_{(i,j),(i',j')}\right]$*, where*

$$a_{(i,j),(i',j')} = \sum_{\substack{k_0+k_1+\ldots+k_c=v_j=(j-a+p(a-j'))/a,\\ pi'=bv_j+ck_c+(c-1)k_{c-1}+\ldots+k_1+i}} \binom{(j-a+p(a-j'))/a}{k_0,\ldots,k_c} m_0^{k_0}m_1^{k_1}\ldots m_c^{k_c}. \tag{5}$$

Proof. Using Lemma 3.1 and $y^a = x^b f(x)$, we find that

$$\begin{aligned}\omega_{i,j} &= \frac{x^{i-1}y^{j-1+(a-1)(p-1)}}{y^{p(a-1)}}\\ &= \frac{x^{i-1}y^{p(u_j-1)+av_j}}{y^{p(a-1)}}\\ &= \left(\frac{y^{u_j-1}}{y^{a-1}}\right)^p \sum_{k_0+k_1+\ldots+k_c=v_j} \binom{v_j}{k_0,\ldots,k_c} m_0^{k_0}m_1^{k_1}\ldots m_c^{k_c}x^{bv_j+ck_c+(c-1)k_{c-1}+\ldots+k_1+i-1}.\end{aligned}$$

Then, using Lemma 2.2,

$$\mathscr{C}(\omega_{i,j}) = \sum_{\substack{k_0+k_1+\ldots+k_c=v_j,\\ p|bv_j+ck_c+(c-1)k_{c-1}+\ldots+k_1+i}} \binom{v_j}{k_0,\ldots,k_c} m_0^{k_0}m_1^{k_1}\ldots m_c^{k_c}\omega_{s_{i,j}(k_0,k_1,\ldots,k_c),u_j}.$$

Now, Lemmas 3.4, 3.5, and 3.6 imply that $(s_{i,j}(k_0,\ldots,k_c),u_j)\in\mathcal{N}$, so $\omega_{s_{i,j}(k_0,\ldots,k_c),u_j}$ is a basis element. The result follows.

□

Using (2), we deduce the following.

Theorem 3.9. *The number of points on the smooth projective model of a superelliptic curve defined by* $y^a = x^b\left(m_cx^c + m_{c-1}x^{c-1} + \ldots + m_0\right)$ *over* $\mathbb{F}_p$ *is*

$$\#C(\mathbb{F}_p) \equiv 1 - \sum_{(i,j)\in N}\left(\sum_{\substack{k_0+k_1+\ldots+k_c=v_j=(p-1)(a-j)/a,\\ p|bv_j+ck_c+(c-1)k_{c-1}+\ldots+k_1+i}} \binom{(p-1)(a-j)/a}{k_0,\ldots,k_c} m_0^{k_0}m_1^{k_1}\ldots m_c^{k_c}\right) \pmod p,$$

where p *is a prime greater than* a *and* $b+c$*.*

Corollary 3.10. *The number of points on the curve* $C : y^a = m_c x^{b+c} + m_0 x^b$ *is*

$$\#C(\mathbb{F}_p) \equiv 1 - \sum_{(i,j)\in N} \left[\begin{matrix} (p-1)(a-j)/a \\ (p-1)(ai+bj-ab)/(ac) \end{matrix} \right] m_0^{k_0} m_c^{k_c} \pmod{p}, \tag{6}$$

where $k_0 = \frac{(p-1)((a-j)(b+c)-ai)}{ac}$ *and* $k_c = \frac{(p-1)(ai+bj-ab)}{ac}$.

Remark 3.11. A rather interesting number-theoretic application of Theorem 3.9 tells us the number of solutions to the Diophantine equation $y^a = h(x) = r_a x^a + r_{a-1} x^{a-1} + \ldots + r_1 x + r_0$ modulo p. Define $\text{ind}_{\zeta_p}(m)$ to be the smallest positive integral power of some primitive root ζ_p of $\mathbb{F}_p$ such that $\zeta_p^{\text{ind}_{\zeta_p}(m)} \equiv m \pmod{p}$. Then, $y^a = h(x)$ has exactly $p + 1 - w(r_a)$ solutions $(x, y) \in \mathbb{F}_p$ if at least one of r_a, r_{a-1} is nonzero in $\mathbb{F}_p$, at least one of r_1, r_0 is nonzero in $\mathbb{F}_p$, $\gcd(a, p-1) = 1$, $p > 16g^2$, and $h(x)$ is square-free, where g is the genus of curve C associated with the Diophantine equation and $w(r_a)$, the number of ath roots of r_a, is given by

$$w(r_a) = \begin{cases} 0, & \text{if } r_a \text{is not a residue of degree } a \text{modulo } p; \\ \gcd\left(\text{ind}_{\zeta_p}(r_a), p\right), & \text{otherwise.} \end{cases}$$

It is worth noting that this can be extended to any Diophantine equation of the form $y^a = x^b \left(m_c x^c + m_{c-1} x^{c-1} + \ldots + m_0\right)$. However, the number of solutions to this equation is not the same as $\#C(\mathbb{F}_p)$: Theorem 3.9 gives the number of points on the smooth projective model of the superelliptic curve defined by this equation, which is usually different from the number of points in the affine model. Thus, for the general case, a tiny correction related to the singular points has to be made to account for the discrepancy between $\#C(\mathbb{F}_p)$ and the number of solutions to the equation $y^a = x^b \left(m_c x^c + m_{c-1} x^{c-1} + \ldots + m_0\right)$.

4 Fast Point-Counting Algorithms

Given our work in Section 3, we have reduced the problem of computing the Hasse-Witt (and hence $\#C(\mathbb{F}_p)$ and $\#J_C(\mathbb{F}_p) \bmod p$) to computing certain multinomial coefficients modulo p.

4.1 Computing $\#C(\mathbb{F}_p)$ for Specific Trinomial Superelliptic Curves

We consider a specific class of superelliptic curves, namely those of the form $y^a = m_c x^{b+c} + m_0 x^b$. To compute the number of points on these specific trinomial superelliptic curves, we generalize Theorem 3.5 (which applies to curves of the form $y^2 = x^8 + c$ and $y^2 = x^7 - cx$) in Fité and Sutherland's paper [7] to apply to this much more general class of curves.

Theorem 4.1. *Given a curve $C : y^a = m_c x^{b+c} + m_0 x^b$ and a prime $p > 16g^2$, where g is the genus of C and $\gcd(ac, p-1) = e = 3, 4, 6,$ or 8, there exists an algorithm that computes $\#C(\mathbb{F}_p)$*

- *probabilistically in $O(\mathrm{M}(\log p)\log p)$ time;*
- *deterministically in $O\left(\mathrm{M}(\log p)\log^2 p \log\log p\right)$ time, assuming GRH;*
- *deterministically in $O\left(\mathrm{M}(\log^3 p)\frac{\log^2 p}{\log\log p}\right)$ time.*

In addition, given a positive integer N we can compute $\#C(\mathbb{F}_p)$ for all primes p, such that $3 < p \leq N$, deterministically in $O\left(N\mathrm{M}\left(\log N\right)\right)$ time.

Proof. We give such an algorithm. It consists of three steps: 1) computing $\begin{bmatrix}(p-1)(a-j)/a \\ (p-1)(ai+bj-ab)/(ac)\end{bmatrix}$ modulo p for all $(i,j) \in \mathcal{N}$, 2) finding $\#C(\mathbb{F}_p) \bmod p$ using (6), and 3) determining $\#C(\mathbb{F}_p)$ using (1). Assuming a and c are fixed, the second step runs in $O(\mathrm{M}(\log p)\log p)$ time and the third step runs in at most $O(\mathrm{M}(\log p))$ time. In what follows, we will describe and determine the time complexity of the first step. We shall then see that, in similar fashion to the proof of Theorem 3.5 in Fité and Sutherland's paper [7], the desired runtimes follow from the time complexities of the first step.

Let $\gcd(ac, p-1) = e$ and $ac = ed, p-1 = ef$, where $d, e, f \in \mathbb{N}$. Note that $\gcd(d, f) = 1$. Then, we have

$$\begin{bmatrix} (p-1)(a-j)/a \\ (p-1)(ai+bj-ab)/(ac) \end{bmatrix} = \begin{bmatrix} (p-1)(ac-jc)/(ac) \\ (p-1)(ai+bj-ab)/(ac) \end{bmatrix} = \begin{bmatrix} (ed-jc)f/d \\ (ai+bj-ab)f/d \end{bmatrix}.$$

Using congruences for binomial coefficients of the form $\binom{rf}{sf}$ modulo primes of the form $p = ef+1$, we can compute $\#C(\mathbb{F}_p)$ extremely quickly. In particular, when $e = 3, 4, 6,$ or 8, it is possible to compute $\binom{rf}{sf}$ modulo p efficiently for $1 \leq s < r \leq e-1$. Note that, by definition,

$$\begin{bmatrix} (ed-jc)f/d \\ (ai+bj-ab)f/d \end{bmatrix} = \begin{cases} \binom{(ed-jc)f/d}{(ai+bj-ab)f/d}, & \text{if } d|ed-jc \text{ and } d|ai+bj-ab; \\ 0, & \text{otherwise.} \end{cases}$$

We can then compute this expression quickly using results from Hudson and Williams' work [14].

We now give explicit congruence relations for the aforementioned binomial coefficients $\binom{rf}{sf}$. We will also use similar notation to that of Hudson and William [14]: $p = a_4^2 + b_4^2$ for $e = 4$, $p = a_3^2 + 3b_3^2$ for $e = 3, 6$, $p = a_8^2 + 2b_8^2$ for $e = 8$, $a_4 \equiv 1 \pmod 4$, $a_3 \equiv 1 \pmod 3$, and $a_8 \equiv 1 \pmod 4$. For each e, define "representative binomial coefficients" as the binomial coefficients sufficient to generate all of the other binomial coefficients through trivial congruences [14]. Table 1 summarizes the congruence relations for the representative binomial coefficients for $e = 3, 4, 6,$ and 8 (let $\overline{b_3} \in \mathbb{Z}$ be $b_3 \bmod 3$).

We have effectively reduced the problem of computing binomial coefficients modulo p to a problem of solving Diophantine equations of the form $x^2 + dy^2 = m$. We can then employ Cornacchia's algorithm [14] to find $a_3, a_4, a_8, b_3, b_4,$ and b_8. However, before doing so, we must find a square root of $-d$ modulo m.

There are three well-known methods of efficiently computing a square root modulo p: the Cipolla-Lehmer, Tonelli-Shanks, and Schoof algorithms [7]. Define $\mathrm{M}(n)$ to be the time needed

$\binom{af}{bf}$	$e=3$	4	6	8
$\binom{2f}{f}$	$\begin{cases} 2a_3 & \text{if } \overline{b_3}=0 \\ -a_3-3b_3 & \text{if } \overline{b_3}=1 \\ -a_3+3b_3 & \text{if } \overline{b_3}=2 \end{cases}$	$2a_4$	$\begin{cases} 2(-1)^f a_3 & \text{if } \overline{b_3}=0 \\ (-1)^f(-a_3+3b_3)b_3 & \text{if } \overline{b_3}=1 \\ (-1)^f(-a_3-3b_3)b_3 & \text{if } \overline{b_3}=2 \end{cases}$	$(-1)^{b_4/4}\,2a_8$
$\binom{3f}{f}$	-	-	$2a_3$	$(-1)^{f+b_4/4}\,2a_4$
$\binom{4f}{f}$	-	-	-	$(-1)^f\,2a_8$
$\binom{5f}{2f}$	-	-	-	$(-1)^{f+b_4/4}\,2a_8$

Table 1: The entry that corresponds to the row with representative binomial coefficient $\binom{af}{bf}$ and column e is congruent to the $\binom{af}{bf} \pmod{ef+1}$. For example, $\binom{3f}{f} \equiv 2a_3 \pmod{6f+1}$.

to compute the product of two n-digit numbers. For practical applications, we can take $\mathrm{M}(n)$ to be $O(n \log n \log\log n)$ or $O\left(n (\log n)\, 2^{\log^* n}\right)$ [4]. The probabilistic Cipolla-Lehmer approach involves factorization of a quadratic over $\mathbb{F}_p[x]$ and takes $O(\mathrm{M}(\log p)\log p)$ time. The deterministic Tonelli-Shanks approach requires computing a generator for the maximal 2-subgroup of $\mathbb{F}_p^*$ and takes $O\left(\mathrm{M}(\log p)\log^2 p \log\log p\right)$ time, assuming the generalized Riemann hypothesis (GRH). The Schoof approach takes advantage of properties of certain elliptic curves and requires $O\left(\mathrm{M}(\log^3 p)\frac{\log^2 p}{\log\log p}\right)$ time. Fité and Sutherland discuss these algorithms in much greater detail in their study of computing $\#C(\mathbb{F}_p)$ for certain hyperelliptic curves [7].

We can now use Cornacchia's algorithm: given a Diophantine equation of the form $x^2+dy^2 = m$ and a square root of $-d$ modulo m, such that $\gcd(d,m)=1$, we can compute a solution (x,y) (if one exists). It turns out that the time complexity of Cornacchia's algorithm is essentially the same as that of the Euclidean algorithm [7].

To compute a_3, a_4, a_8, b_3, b_4, and b_8, we can first use Cornacchia's algorithm to find a solution (x,y) to $x^2+dy^2=p$, where d is $1, 2$, or 3. Taking the equation $x^2+y^2=p$ modulo 4, it is clear that at least one of x and y is odd. Without loss of generality, say x is odd. We can replace x with $-x$, so we can assume $x \equiv 1 \pmod 4$ and let $a_4 = x$. Likewise, if we take $x^2+3y^2=p$ modulo 3, we can set $a_3 \equiv 1 \pmod 3$, since x is either 1 or -1 modulo 3 and if we take $x^2+2y^2=p$ modulo 2, x must be odd, so we can set $a_8 \equiv 1 \pmod 4$ by letting a_8 equal x or $-x$ modulo 4. $\square$

Remark 4.2. It is worth noting that Pila's generalization of Schoof's algorithm can compute $\#C\left(\mathbb{F}_p\right)$ and the zeta function in time polynomial in $\log p$; however, the time complexity is a large power of $\log p$ that grows quickly with the genus. Even in genus 2, the most efficient algorithms using Pila's approach take $O\left(\log^{6+o(1)} p\right)$ time [17], but the algorithm we have just described performs much better.

Example 4.3. Say $p = 564819669946735512444543556507$ (99-bit) and $C : y^4 = x^{11} + x^8$ (note that $\gcd(564819669946735512444543556506, 12) = 6$. The algorithm given in Theorem 4.1 computed

$$\#C\left(\mathbb{F}_p\right) = 564819669946736263601275822996$$

in 66.2 ms using SageMath Version 7.3 on an Intel Celeron 2955U processor at 1.4 GHz. In comparison, a brute-force algorithm to compute the number of points for the same curve with $p = 10133$ (14-bit) instead took over six hours to run.

It should be noted that this algorithm can be extended to find $\#J_C\left(\mathbb{F}_p\right)$ for genus two and three curves of the same form, with the additional condition that $a|p-1$. It turns out that Hasse-Witt matrix $A_p\left(C\right)$ of such a curve is diagonal; it is not hard to see that $s_{i,j}\left(k_0, \ldots, k_c\right)$ can only be i and u_j can only be j. If $A_p\left(C\right)$ is diagonal, $A_p\left(C\right) - I$ is also diagonal, so (3) tells us that

$$\#J_C\left(\mathbb{F}_p\right) \equiv (-1)^g \prod_{(i,j)\in\mathcal{N}} \left(\begin{bmatrix} (p-1)(a-j)/a \\ (p-1)(ai+bj-ab)/(ac) \end{bmatrix} m_0^{k_0} m_c^{k_c} - 1 \right) \pmod{p}, \qquad (7)$$

where $k_0 = \frac{(p-1)((a-j)(b+c)-ai)}{ac}$ and $k_c = \frac{(p-1)(ai+bj-ab)}{ac}$.

For genus two curves, the Hasse-Weil bounds and $\#J_C\left(\mathbb{F}_p\right)$ modulo p gives us a maximum of five candidates for $\#J_C\left(\mathbb{F}_p\right)$, so we can compute $\#J_C\left(\mathbb{F}_p\right)$ in the same time as that of $\#C\left(\mathbb{F}_p\right)$, because arithmetic in the Jacobian has a comparatively lower time complexity [8, 18]. For genus three curves, the Hasse-Weil bounds and $\#J_C\left(\mathbb{F}_p\right)$ modulo p gives us a maximum of $\left\lceil 40\sqrt{p} \right\rceil$ candidates for $\#J_C\left(\mathbb{F}_p\right)$, so we can compute $\#J_C\left(\mathbb{F}_p\right)$ in $O\left(\sqrt{p}\right)$ time [18, 19]. In comparison, Harvey

has written down an improvement of Kedlaya's point-counting algorithm to compute $\#J_C(\mathbb{F}_p)$ for hyperelliptic curves in $O\left(p^{1/2+o(1)}\right)$ time [11]. Our algorithm, however, is not efficient for curves of higher genus.

4.2 Computing $\#C(\mathbb{F}_p)$ and $A_p(C)$ for General Superelliptic Curves

We describe an algorithm to simplify the entries of the Hasse-Witt matrix modulo p for a general superelliptic curve and compute $\#C(\mathbb{F}_p)$ and $A_p(C)$ efficiently. More generally, Harvey has shown how to compute the zeta function of any algebraic variety over $\mathbb{F}_p$ (including superelliptic curves) in $O(\sqrt{p}\log^{2+\epsilon} p)$ time [10], using Bostan et al.'s algorithm [2]. To do so, Harvey describes a new trace formula and a deformation recurrence; on the other hand, we find the zeta function modulo p by reducing the Hasse-Witt matrix modulo p.

Throughout this section, we will assume that $b = 0$, i.e. the equation of superelliptic curve is $y^a = f(x) = m_c x^c + \ldots + m_0$, where $m_0 \neq 0$ (note that we can perform a simple coordinate change to (4)). We summarize the algorithm as follows.

Theorem 4.4. *Given a curve $C : y^a = m_c x^c + \ldots + m_0$ and a prime $p > 16g^2$, where g is the genus of C, there exists an algorithm that computes $\#C(\mathbb{F}_p)$ in $O\left(\mathrm{M}\left(\sqrt{p}\log p\right)\right)$ time, assuming fixed a and c.*

Proof. We give such an algorithm. It consists of three steps: 1) adapting Bostan et al.'s algorithm [2] to simplify the entries of the Hasse-Witt matrix, 2) adding up the diagonal entries of the Hasse-Witt matrix to find $\#C(\mathbb{F}_p) \bmod p$, and 3) determining $\#C(\mathbb{F}_p)$ using (1). The second step runs in $O(\mathrm{M}(\log p)\log p)$ time and the third step runs in at most $O(\mathrm{M}(\log p))$ time. In what follows, we will describe the first step and determine its time complexity.

In Section 3, we found an explicit formula for the Hasse-Witt matrix of any superelliptic curve. Specifically, (5) tells us that the problem of simplifying the Hasse-Witt matrix is essentially a question of simplifying certain multinomial coefficients modulo p. We can reinterpret this as a

problem of computing coefficients of certain powers of x in f^{v_j}, where v_j is defined in Lemma 3.4. For convenience, let $f_{i,j}$ denote the coefficient of x^i in f^j.

Setting $b = 0$ in (5) and using the multinomial theorem (note that $pi' - i = bv_i + ck_c + (c-1)k_{c-1} + \ldots + k_1$) imply that

$$a_{(i,j),(i',j')} = f_{pi'-i,v_j},$$

where $a_{(i,j),(i',j')}$ is the coefficient of $\omega_{(i',j')}$ when $\omega_{(i,j)}$ is written as a linear combination of the basis of regular differentials.

We can now use Bostan et al.'s algorithm [2] for quickly computing coefficients of large powers of a polynomial. In their paper, they apply their algorithm to point-counting for hyperelliptic curves. We generalize their method to superelliptic curves; the extension is quite similar to the original method for hyperelliptic curves, so we will omit the finer details [1]. We can say that

$$f\left(f^{v_j}\right)' - v_j f' f^{v_j} = 0.$$

Comparing coefficients of x^k on both sides, we find that

$$(k+1)\, m_0 f_{k+1,v_j} + (k - v_j)\, m_1 f_{k,v_j} + \ldots + ((k-c+1) - cv_j)\, f_{k-c+1,v_j} = 0.$$

Let

$$A(k) = \begin{pmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ddots & 1 \\ \frac{m_c(cv_j-(k-c))}{m_0 k} & \frac{m_{c-1}((c-1)v_j-(k-c+1))}{m_0 k} & \ldots & \ldots & \frac{m_1(v_j-(k-1))}{m_0 k} \end{pmatrix}.$$

Now, if we define the vector $U_k = \left[f_{k-c+1,v_j}, \ldots, f_{k-1,v_j}, f_{k,v_j}\right]^t$ and initial vector $U_0 = \left[0, \ldots, 0, m_0^{v_j}\right]^t$, we have $U_k = A(k)U_{k-1}$ for $k \in \mathbb{N}$. Thus, $U_k = A(k)A(k-1)\ldots A(1)U_0$. Thus, to compute $a_{(i,j),(i',j')} = f_{pi'-i,v_j}$, it suffices to find $U_{pi'-1}$ (note that $f_{pi'-i}$ is the ith en-

try of $U_{pi'-1}$ from the right). Bostan et al.'s algorithm [2] cannot be applied directly because the entries of $A(k)$ are rational functions, so we must modify the U_i slightly: let $V_k = m_0^k k! U_k$ and $B(k) = m_0 k A(k)$. Then, $V_k = B(k)B(k-1)\dots B(1)V_0$ and the entries of the matrix $B(k)$ are polynomials in $\mathbb{F}_p[k]$.

Using either of the algorithms from Theorems 14 and 15 from Bostan et al.'s paper [2], we can compute $U_{pi'-1}$ for all i' and a particular j in $O\left(\mathrm{M}\left(\sqrt{p}\log p\right)\right)$ time. The desired runtime follows from applying this repeatedly to all j (there are at most a possible values of j because $1 \leq j \leq a$). □

Remark 4.5. Given that $p > g$, the proof of Theorem 4.4 also shows that we can simplify $A_p(C)$, the Hasse-Witt matrix modulo p, in $O\left(\mathrm{M}\left(\sqrt{p}\log p\right)\right)$ time.

It should be noted that this algorithm can be extended to find $\#J_C(\mathbb{F}_p)$ for genus two and three curves of the same form. We can compute $\#J_C(\mathbb{F}_p)$ in $O\left(\mathrm{M}\left(\sqrt{p}\log p\right)\right)$ time for genus two curves and in $O\left(\mathrm{M}\left(\sqrt{p}\log p\right)\right)$ time for genus three curves (refer to the discussion at the end of Subsection 4.1).

5 Conclusion

In this paper, we employed the Cartier operator to derive the entries of the Hasse-Witt matrix of a superelliptic curve in terms of multinomial coefficients, as shown in Theorem 3.8. We then described and determined the time complexities of efficient point-counting algorithms for both specific trinomial and general curves. In particular, for specific trinomial superelliptic curves, we reduced the problem of point-counting to a problem of solving Diophantine equations of the form $x^2 + dy^2 = m$ and computed $\#C(\mathbb{F}_p)$ probabilistically in $O(\mathrm{M}(\log p)\log p)$ time, deterministically in $O\left(\mathrm{M}(\log p)\log^2 p \log\log p\right)$ (assuming the generalized Riemann hypothesis), and deterministically in $O\left(\mathrm{M}(\log^3 p)\frac{\log^2 p}{\log\log p}\right)$ time, assuming $p > 16g^2$ and $\gcd(ac, p-1) = 3, 4, 6$ or 8. We also described how to compute $\#J_C(\mathbb{F}_p)$ with the same running times for $g = 2$ and with running time

$O\left(\sqrt{p}\right)$ if $g = 3$, assuming that a also divides $p - 1$. As shown in Example 4.3, we implemented and demonstrated the efficiency of the algorithm given in Theorem 4.1 to compute $\#C\left(\mathbb{F}_p\right)$ for the superelliptic curve with affine model $y^4 = x^{11} + x^8$ over $\mathbb{F}_{564819669946735512444543556507}$. For general superelliptic curves, we employed Bostan et al.'s algorithm [2] to compute $\#C\left(\mathbb{F}_p\right)$, $A_p\left(C\right)$, and $\#J_C\left(\mathbb{F}_p\right)$ in $O\left(\mathrm{M}\left(\sqrt{p}\log p\right)\right)$ time, assuming $p > 16g^2$ (and additionally $g = 2$ or 3 for $\#J_C\left(\mathbb{F}_p\right)$). To the best of our knowledge, we have written down the fastest point-counting algorithms for specific trinomial superelliptic curves.

In the future, we hope to write down explicit algorithms for computing $\#J_C\left(\mathbb{F}_p\right)$, which will plausibly involve finding the coefficients of the characteristic polynomial of Frobenius. We also plan on implementing the algorithm given in Theorem 4.4 through a simple modification of the original algorithm described by Bostan et al. for hyperelliptic curves [2]. We also hope to improve the efficiencies of our algorithms. Harvey et al. discussed how the characteristic polynomials of Frobenius for genus three hyperelliptic curves can be computed in $p^{1/4+o(1)}$ time by "lifting" the characteristic polynomials modulo p [12]; it may be possible to extend these methods to the curves of interest in this paper. For studying Sato-Tate distributions, we will also look into extending Harvey et al.'s approach for efficiently computing $\#C\left(\mathbb{F}_p\right)$ and Hasse-Witt matrices for many p at once [13]. Finally, we hope to extend our methods to non-superelliptic curves. Although not discussed in this paper, we have also looked into interpreting the Hasse-Witt matrix through the lens of Cech cohomology. We believe that such methods are easier to generalize, especially for trinomial curves of the form $y^{m_1} = c_1 x^{n_1} + c_2 x^{n_2} y^{m_2}$.

6 Acknowledgments

I would like to express my deepest gratitude and appreciation to my mentor Nicholas Triantafillou, an MIT PhD student, for his support, dedication, and guidance in my research. I would also like to thank Professor Andrew Sutherland from MIT for suggesting this project and for providing valuable

advice. Finally, none of this research would have been possible without the help of Dr. Tanya Khovanova, Professor Pavel Etingof, and Dr. Slava Gerovitch from MIT for organizing PRIMES-USA, a math research program for high school juniors.

References

[1] M. Bauer, E. Teske, and A. Weng, Point counting on Picard curves in large characteristic, *Math. Comp.* **74** (2005), 1983–2005.

[2] A. Bostan, P. Gaudry, and E. Schost, Linear recurrences with polynomial coefficients and application to integer factorization and Cartier-Manin operator, *SIAM J. Comput.* **36** (2007), 1777–1806.

[3] A. Couvreur, Codes and the Cartier operator, *Proc. Amer. Math. Soc.* **142** (2014), 1983–1996.

[4] A. De, P. P. Kurur, C. Saha, and R. Saptharishi, Fast integer multiplication using modular arithmetic, *SIAM J. Comput.* **42** (2013), 685–699.

[5] B. Deconinck and M. Patterson, Computing with plane algebraic curves and Riemann surfaces: the algorithms of the Maple package "algcurves", in A. I. Bobenko and C. Klein, eds., *Computational Approach to Riemann Surfaces*, Lect. Notes Math., Vol. 2013, Springer, 2011, pp. 67–123.

[6] F. Fité, K. S. Kedlaya, V. Rotger, and A. V. Sutherland, Sato–Tate distributions and Galois endomorphism modules in genus 2, *Compos. Math.* **148** (2012), 1390–1442.

[7] F. Fité and A. V. Sutherland, Sato-Tate groups of $y^2 = x^8 + c$ and $y^2 = x^7 - cx$, *Contemp. Math.* **663** (2016), 103–126.

[8] E. Furukawa, M. Kawazoe, and T. Takahashi, Counting points for hyperelliptic curves of type $y^2 = x^5 + ax$ over finite prime fields, in *SAC 2003*, Lect. Notes in Comp. Sci., Vol. 3006, Springer, 2003, pp. 26–41.

[9] A. Garcia and S. Tafazolian, Certain maximal curves and Cartier operators, *Acta Arith.* **135** (2007), 199–218.

[10] D. Harvey, Computing zeta functions of arithmetic schemes, *Proc. Lond. Math. Soc.* **111** (2015), 1379–1401.

[11] D. Harvey, Kedlaya's algorithm in larger characteristic, *Int. Math. Res. Not.* (2007).

[12] D. Harvey, M. Massierer, and A. V. Sutherland, Computing L-series of geometrically hyperelliptic curves of genus three, preprint, https://arxiv.org/abs/1605.04708.

[13] D. Harvey and A. V. Sutherland, Computing Hasse-Witt matrices of hyperelliptic curves in average polynomial time II, *Contemp. Math.* **663** (2016), 136–154.

[14] R. H. Hudson and K. S. Williams, Binomial coefficients and Jacobi sums, *Trans. Amer. Math. Soc.* **281** (1984), 431–505.

[15] N. Koblitz, Why study equations over finite fields?, *Math. Mag.* **55** (1982), 144–149.

[16] Y. I. Manin, The Hasse-Witt matrix of an algebraic curve, *Amer. Math. Soc. Transl. Ser. 2* **45** (1965), 245–264.

[17] J. Pila, Frobenius maps of abelian varieties and finding roots of unity in finite fields, *Math. Comp.* **55** (1990), 745–763.

[18] B. Poonen, Computational aspects of curves of genus at least 2, in *ANTS II*, Lect. Notes in Comp. Sci., Vol. 1122, Springer, 1996, pp. 283–306.

[19] G. Sohn and and H. Kim, Explicit bounds of polynomial coefficients and counting points on Picard curves over finite fields, *Math. Comput. Model.* **49** (2009), 80–87.