Hyperelliptic Curves and Cryptography

Michael Jacobson, Jr.
Department of Computer Science
University of Calgary

Alfred Menezes
Department of Combinatorics & Optimization
University of Waterloo

Andreas Stein
Department of Mathematics
University of Illinois at Urbana-Champaign

Dedicated to Hugh Williams on the occasion of his 60th birthday.

Abstract. In 1989, Koblitz proposed using the jacobian of a hyperelliptic curve defined over a finite field to implement discrete logarithm cryptographic protocols. This paper provides an overview of algorithms for performing the group law (which are necessary for the efficient implementation of the protocols), and algorithms for solving the hyperelliptic curve discrete logarithm problem (whose intractability is essential for the security of the protocols). Also considered are destructive applications of hyperelliptic curves—solving instances of the elliptic curve discrete logarithm by using the technique of Weil descent to reduce them to instances of the hyperelliptic curve discrete logarithm problem.

1 Introduction

Since the discovery of public-key cryptography in 1975 by Diffie, Hellman and Merkle, a variety of finite cyclic groups have been used to design public-key cryptographic schemes. Let G be a finite cyclic group of order n, and let α be a generator of G. The group parameters α and n are public knowledge. Each user A selects a random integer a from the interval $[0, n-1]$ and computes α^a. A publishes α^a as her public key, and keeps a secret. In the basic Diffie-Hellman key agreement protocol [14], users A and B exchange their public keys α^a and α^b over an unsecured channel and then compute $K = \alpha^{ab}$ as their shared secret. ElGamal [15]

1991 *Mathematics Subject Classification.* 11R58, 11Y16, 11Y65.
The first author's research was supported by NSERC Discovery Grant #238981-01.
The second author's research was supported by NSERC Discovery Grant #203813-02.
The third author's research was supported by NSF Grant DMS-0201337.

devised public-key encryption schemes whereby anyone can use α^a to encrypt messages which can only be decrypted by A, and signature schemes whereby A uses her private key a to generate signatures on messages whose validity can be verified by anyone using A's public key α^a.

Cryptographically suitable groups. There are three fundamental constraints that determine which groups G are suitable for practical implementation of these cryptographic protocols. First, the group elements should have a compact representation. Preferably, each group element should be representable as a unique bit string of length approximately $\log_2 n$. Second, given a representation for the elements, an efficient algorithm should be known for performing the group operation. Then, exponentiation can be efficiently performed by the repeated-square-and-multiply algorithm. Lastly, in order to preserve the secrecy of the private key a, it should be computationally infeasible for anyone to compute a given only α, n and α^a. This problem is known as the *discrete logarithm problem* (DLP) in G.

Abelian varieties over finite fields provide an abundant source of finite groups that could potentially satisfy these constraints. Roughly speaking, an abelian variety over a finite field $\mathbb{F}_q$ is the set of solutions to a system of multivariate equations defined over $\mathbb{F}_q$ with an accompanying group law given by rational formulae. The abelian varieties most suitable for cryptographic applications appear to be the jacobians of algebraic curves. Several families of curves have been proposed including elliptic curves [43, 52], hyperelliptic curves [44], Picard curves [7], superelliptic curves [27], and C_{ab} curves [5]. Efficient algorithms for performing the group operation in the jacobians of these curves are known; these algorithms are much more efficient that the algorithms known for arithmetic in the jacobians of general algebraic curves [39, 83].

Known algorithms for the discrete logarithm problem. Pohlig and Hellman [62] observed that an instance of the DLP in G can be efficiently reduced to instances of the DLP in prime-order subgroups of G. The best algorithm known for solving the DLP is Pollard's rho algorithm [63] and its parallelization by van Oorschot and Wiener [55]. Pollard's algorithm has a purely exponential expected running time of $\sqrt{(\pi n/2)}$ group operations and negligible storage requirements. It can be easily implemented on a network of r processors with an r-fold speedup. Moreover, the processors do not communicate with each other. Nechaev [54] and Shoup [71] established a lower bound of $\Omega(\sqrt{n})$ for the DLP in *generic* groups, thus showing that Pollard's rho algorithm is essentially the best generic algorithm possible for the DLP. In order to provide maximum resistance to the Pohlig-Hellman and Pollard's rho algorithms, the group order n must have a large prime factor (e.g., greater than 2^{160}), and preferably should itself be prime.

Algorithms for the DLP that are faster than Pollard's rho method are known for some families of groups. The most powerful is the index-calculus method which yields subexponential-time algorithms for the DLP in some groups including the multiplicative group of a finite field [13, 35, 2], the class group of an imaginary quadratic number field [37], and the jacobian of a high-genus hyperelliptic curve [3].

If C is an algebraic curve defined over a finite field $\mathbb{F}_q$, then the Tate pairing can be used to efficiently reduce the DLP in the jacobian $J_C(\mathbb{F}_q)$ to the DLP in the multiplicative group of an extension field $\mathbb{F}_{q^k}$ [24]. The extension degree k is the smallest positive integer for which $\#J_C(\mathbb{F}_q)$ (or the largest prime factor of

$\#J_C(\mathbb{F}_q)$) divides $q^k - 1$. For most curves defined over $\mathbb{F}_q$, it is expected that k is too large. However, for some curves C, k is indeed small and hence the Tate pairing reduction yields a subexponential-time algorithm for the DLP in $J_C(\mathbb{F}_q)$. For example, if C is a supersingular elliptic curve defined over a finite field, then $k \in \{1, 2, 3, 4, 6\}$.

If C is an algebraic curve defined over a prime field $\mathbb{F}_p$ such that $\#J_C(\mathbb{F}_p) = p$, then the DLP in $J_C(\mathbb{F}_p)$ can be efficiently solved [66, 69, 74, 64].

The curves that appear to be the most attractive for cryptographic applications are low-genus hyperelliptic curves. This is because very efficient algorithms are known for the group law and because of the absence of subexponential-time algorithms for the discrete logarithm problem for carefully chosen curves. Genus 1 curves (i.e., elliptic curves) are especially attractive and have been widely standardized and deployed in practice. Genus 2 and 3 hyperelliptic curves have also received significant attention in recent years, and several research groups have reported software and hardware implementations.

Destructive applications of hyperelliptic curves. In 2000, Gaudry, Hess and Smart (GHS) [34] showed how the elliptic curve discrete logarithm problem (ECDLP) for an elliptic curve over a characteristic two finite field $\mathbb{F}_{2^n}$ can be reduced to the DLP in the jacobian of a hyperelliptic curve C defined over a subfield of $\mathbb{F}_{2^n}$. For some elliptic curves, the genus of C is small and the GHS reduction yields an algorithm for solving the ECDLP instance that is faster than Pollard's rho algorithm. Therefore, efficient algorithms for performing the group law in the jacobian of hyperelliptic curves and for solving the hyperelliptic curve discrete logarithm problem (HCDLP) are also of interest because they can be used to attack elliptic curve cryptosystems.

Organization. This paper provides an overview of algorithms for performing the group law and for solving the discrete logarithm problem in the jacobian of hyperelliptic curves. The remainder of the paper is organized as follows. Section 2 presents an introduction to hyperelliptic curves and describes algorithms for performing the group law. Index-calculus methods for the HCDLP are studied in Section 3. In Section 4, we outline the technique of Weil descent for reducing the ECDLP to the HCDLP and discuss its cryptographic implications. Section 5 surveys implementations of index-calculus algorithms for the HCDLP. Finally, some research problems in hyperelliptic curve cryptography are listed in Section 6.

2 Hyperelliptic curve arithmetic

We provide a brief introduction to the arithmetic of hyperelliptic curves. For a more detailed (but elementary) exposition, see [51, 77].

2.1 Basic definitions. Let $k = \mathbb{F}_q$ be a finite field with q elements, and let $\overline{k} = \bigcup_{n\geq 1} \mathbb{F}_{q^n}$ be its algebraic closure. A non-singular (imaginary quadratic) *hyperelliptic curve C of genus g over k* is defined by an equation of the form

$$C : v^2 + h(u)v = f(u) \quad ,$$

where $h, f \in k[u]$, f is monic, $\deg f = 2g + 1$, $\deg h \leq g$, and if $y^2 + h(x)y = f(x)$ for $(x, y) \in \overline{k} \times \overline{k}$, then $2y + h(x) \neq 0$ or $h'(x)y - f'(x) \neq 0$. We note that C is absolutely irreducible, i.e. irreducible over $\overline{k}$. For any subfield K of $\overline{k}$ containing k, the set

$$C(K) = \{(x,y) : x,y \in K,\ y^2 + h(x)y = f(x)\} \cup \{\infty\}$$

is called the *set of K-rational points on C*. K is called the *base field*. The point ∞ is called the *point at infinity* and corresponds to the only projective point at infinity that satisfies the homogenized equation. For $P = (x,y) \in C(K)$, the *opposite* of P is $\widetilde{P} = (x, -y - h(x)) \in C(K)$; we also define $\widetilde{\infty} = \infty$. A point P on C, also written $P \in C$, is a point $P \in C(\overline{k})$.

For hyperelliptic curves of genus $g \geq 2$, there is no natural group law on $C(K)$. A group law is defined via the jacobian of C over k which is a finite abelian group. A *divisor* on C is a finite formal sum of points on C, $D = \sum_{P \in C} m_P P$, where the $m_P \in \mathbb{Z}$ are 0 for all but finitely many P. Then the *degree* of D is defined by $\deg D = \sum_P m_P$. D is said to be *defined over* K, if $D^\sigma = \sum_P m_P P^\sigma = D$ for all automorphisms σ of $\overline{k}$ over K, where $P^\sigma = (\sigma(x), \sigma(y))$ if $P = (x,y)$, and $\infty^\sigma = \infty$. The set $Div_C(K)$ of divisors of C defined over K forms an (additive) abelian group under the addition rule $\sum m_P P + \sum n_P P = \sum (m_P + n_P) P$. The set $Div_C^0(K)$ of all degree zero divisors of C defined over K is a subgroup of $Div_C(K)$.

The *greatest common divisor* of $D_1 = \sum_P m_P P$ and $D_2 = \sum_P n_P P$ in $Div_C^0(K)$ is defined by $\gcd(D_1, D_2) = \sum_P a_P P - (\sum_P a_P)\infty$, where $a_P = \min(m_P, n_P)$ for $P \in C$. If $K[C] = K[u,v]/(v^2 + h(u)v - f(u))$ denotes the *coordinate ring* of C over K, then the field of fractions $K(C)$ of $K[C]$ is called the *function field* of C over K. For a polynomial $G \in K[C]$, the divisor of G is defined by $\operatorname{div}(G) = \sum_P \operatorname{ord}_P(G) P$, where $\operatorname{ord}_P(G)$ is the order of vanishing of G at P. The divisor of a rational function $F = G/H$ is called a *principal divisor* and is defined by $\operatorname{div}(F) = \operatorname{div}(G) - \operatorname{div}(H)$. We denote by $\mathcal{P}_C(K) = \{\operatorname{div}(F) : F \in K(C)\}$ the *group of principal divisors of C over K*. In particular, we call $\mathcal{P}(C) = \mathcal{P}_C(\overline{k})$ the *group of principal divisors* of C. Since every principal divisor has degree 0, $\mathcal{P}_C(K)$ is a subgroup of $Div_C^0(K)$. Finally, the *jacobian of C over K* is defined by $J_C(K) = Div_C^0(K)/\mathcal{P}_C(K)$. Similarly, one defines the *jacobian of C* by $J(C) = Div^0(C)/\mathcal{P}(C)$. If D_1 and D_2 are degree 0 divisors in the same equivalence class of divisors in $J_C(K)$ we write $D_1 \sim D_2$. We let $[D]$ be the equivalence class of $J_C(K)$ containing the degree 0 divisor D. If the base field K is a finite field with cardinality q^n, then $J_C(K)$ is a finite abelian group and a theorem of Weil's implies that $(\sqrt{q^n} - 1)^{2g} \leq \#J_C(k) \leq (\sqrt{q^n} + 1)^{2g}$. In particular, we see that $\#J_C(K) \approx q^{ng}$ and $\#J_C(k) \approx q^g$.

2.2 Reduced divisors. Let C_f be the set $C(\overline{k}) \setminus \{\infty\}$ of finite points on C. A degree zero divisor $D = \sum_{P \in C_f} m_P P - \left(\sum_{P \in C_f} m_P\right)\infty$ is called *semi-reduced*, if D satisfies for all $P \in C_f$:

(i) $m_P \geq 0$
(ii) if $P \neq \widetilde{P}$ and $m_P > 0$, then $m_{\widetilde{P}} = 0$.
(iii) if $P = \widetilde{P}$ and $m_P > 0$, then $m_P = 1$.

A semi-reduced divisor D is called *reduced*, if D satisfies the additional condition

(iv) $\sum_{P \in C_f} m_P \leq g$.

Reduced divisors have the crucial property that for each $D \in Div_C^0(K)$ there exists a unique reduced divisor D_r such that $[D] = [D_r]$. That means each equivalence class contains a unique reduced divisor and the set of reduced divisors of C over K forms a complete system of representatives for the jacobian of C over K.

Each semi-reduced divisor $D = \sum_{P\in C_f} m_P P - \left(\sum_{P\in C_f} m_P\right)\infty$ defined over K can be uniquely represented by a pair of polynomials $a, b \in K[u]$, where $a(u) = \prod_{P\in C_f}(u - x_P)^{m_P}$ is monic, and b is the unique polynomial such that

(i) $\deg b < \deg a$.
(ii) for all $P \in C_f$: if $m_P \neq 0$, then $b(x_P) = y_P$.
(iii) a divides $b^2 + bh - f$.

In this case, $D = \gcd(\mathrm{div}(a), \mathrm{div}(b - v))$, and we write $D = \mathrm{div}(a, b)$. Therefore, each reduced divisor D defined over K has a unique representation of the form $D = \mathrm{div}(a, b)$, where $a, b \in K[u]$ with a monic, $\deg b < \deg a \leq g$, and $a|(b^2+bh-f)$. The *degree* of D is $\deg a$. Notice that the opposite of $D = \mathrm{div}(a, b)$ is given by $-D = \mathrm{div}(a, -h - b)$. Cantor's algorithm [12] can be used to efficiently compute the sum of two reduced divisors in $J_C(K)$, and express the sum in reduced form. We explain the method and improvements in the subsequent sections.

2.3 The Frobenius endomorphism of the jacobian. We suppose that $C: v^2 + h(u)v = f(u)$ is a hyperelliptic curve of genus g over k in imaginary quadratic form, and that K is a subfield of $\overline{k}$ containing k. Let $\phi : \overline{k} \to \overline{k}$ be the *Frobenius automorphism* defined by $x \mapsto x^q$. The map ϕ extends to $C(\overline{k})$ by $(x, y) \mapsto (x^\phi, y^\phi)$ and $\infty^\phi \mapsto \infty$, and to $Div^0(C)$ by $D = \sum m_P P \mapsto D^\phi = \sum m_P P^\phi$. Thus, ϕ naturally induces an endomorphism $\phi : J(C) \to J(C)$.

This has important practical consequences for the computation of D^ϕ. Let $D = \mathrm{div}(a, b)$ be a reduced divisor defined over K. Then $D^\phi = \mathrm{div}(a^\phi, b^\phi)$ is a reduced divisor defined over K. In particular, if $a = u^l + \sum_{i=0}^{l-1} a_i u^i$ and $b = \sum_{i=0}^{l-1} b_i u^i$, then $a^\phi = u^l + \sum_{i=0}^{l-1} a_i^q u^i$ and $b^\phi = \sum_{i=0}^{l-1} b_i^q u^i$. The computation of D^ϕ requires at most $2g$ exponentiations of elements in K by q. Furthermore, if we use a normal basis representation for elements in $K = \mathbb{F}_{q^n}$, then a^ϕ and b^ϕ can be determined by simply shifting the normal basis representation of each coefficient a_i and b_i. The complexity of computing D^ϕ is therefore at most $2g$ cyclic shifts. Even if a polynomial basis representation is used, the computation of D^ϕ is much less expensive than performing an addition or doubling in $J_C(K)$.

2.4 The extended Euclidean algorithm. We briefly recall the extended Euclidean algorithm for polynomials and its complexity, since it is fundamental for the composition step in the addition of divisors in the jacobian. These results are well-known (see e.g. [30]). Here, let K be any field. Since $\gcd(a_1, a_2, \ldots, a_r) = \gcd(a_1, \gcd(a_2, \ldots, a_r))$, we restrict our attention to the case $r = 2$. Given two polynomials $a, b \in K[u]$, where $\deg a \geq \deg b$, we want to compute polynomials $d, s, t \in K[u]$ such that $d = \gcd(a, b) = sa+tb$. In the extended Euclidean algorithm, we compute polynomials r_i, s_i, $t_i \in K[u]$ for $i = 0, 1, \ldots, n$ as follows. Put $r_0 = a$ and $r_1 = b$. For $i = 2, \ldots, n$, we use division with remainder to compute $q_{i-1}, r_i \in K[u]$ such that $r_{i-2} = q_{i-1}r_{i-1} + r_i$ and $\deg r_i < \deg r_{i-1}$. Here, n is minimal such that $r_{n+1} = 0$ which implies that $r_n = \gcd(a, b)$. In addition, we put $s_0 = 1$, $s_1 = 0$, $t_0 = 0$, $t_1 = 1$, and, for $i = 2, \ldots, n$, we let $s_i = s_{i-2} - q_{i-1}s_{i-1}$, $t_i = t_{i-2} - q_{i-1}t_{i-1}$. Then $s = s_n$ and $t = t_n$. The computation of r_n, s_n, t_n can be done in $O((\deg a)^2)$ operations in K. The half-extended Euclidean algorithm computes d and s (or t) only, i.e. in the Euclidean algorithm, it is enough to determine the r_i's and s_i's (or t_i's).

2.5 Cantor's algorithm. Cantor [12] presented the following algorithm to compute the group law in $J_C(K)$ for any field K, where C is a hyperelliptic curve in imaginary quadratic form. Let $D_1 = \mathrm{div}(a_1, b_1)$ and $D_2 = \mathrm{div}(a_2, b_2)$ be two reduced divisors defined over K. The goal is to find the unique reduced divisor equivalent to $D_1 + D_2$. In order to do so, one performs two steps:

1. **Composition**: Determine a semi-reduced divisor D representing the sum of D_1 and D_2.
2. **Reduction**: Transform the semi-reduced divisor D into a reduced one, i.e. compute D' such that D' is the unique reduced divisor equivalent to D.

For the composition step, it is enough to assume D_1 and D_2 to be semi-reduced. Thus, it is valid for reduced divisors in particular. Given two semi-reduced divisors $D_1 = \mathrm{div}(a_1, b_1)$ and $D_2 = \mathrm{div}(a_2, b_2)$ defined over K, we perform the following steps to determine a semi-reduced divisor $D = \mathrm{div}(a, b)$ defined over K such that $D \sim D_1 + D_2$ on $J_C(K)$.

1. Use the extended Euclidean algorithm to compute $d, x_1, x_2, x_3 \in K[u]$ such that $d = \gcd(a_1, a_2, b_1 + b_2 + h) = x_1a_1 + x_2a_2 + x_3(b_1 + b_2 + h)$.
2. Put $a := a_1a_2/d^2$ and $b := (x_1a_1b_2 + x_2a_2b_1 + x_3(b_1b_2 + f))/d \bmod a$.

Notice that if D_1 and D_2 are reduced divisors, i.e. $\deg a_1, \deg a_2 \leq g$, then $\deg a \leq 2g$. In fact, we then expect in most of the cases that $\deg a = 2g$.

Well known methods for reduction of binary quadratic forms yield the following reduction algorithm. We compute polynomials $a_i, b_i \in K[u]$ for $i = 0, 1 \ldots, l$, where l is minimal such that $\deg a_l \leq g$. Hereby, we put $a_0 := a$, $b_0 := b$, and for $i \geq 1$, we put

$$a_i := (f - hb_{i-1} - b_{i-1}^2)/a_{i-1} \quad , \quad b_i := (-h - b_{i-1}) \bmod a_i \ .$$

We denote by $M(m)$ the number of operations in K for multiplying two polynomials of degree at most m. Then the extended Euclidean algorithm for two polynomials of degree at most m has a complexity of $O(M(m) \log m)$ operations in K (see [30]). The standard algorithms for polynomial multiplication yield $M(m) = O(m^2)$, whereas Karatsuba's method gives $M(m) = O(m^{1.58\cdots})$, and the fast Fourier transformation (FFT) would improve this to $M(m) = O(m \log m \log \log m)$. The latter method mainly applies in a situation where the genus is very large. Thus, the composition of two reduced divisors has a complexity of $O(M(g) \log g)$, whereas the complexity of the reduction algorithm is $O(gM(g))$ operations in K. With standard algorithms for polynomial multiplication, the composition takes $O(g^2)$ operations in K and the reduction has a complexity of $O(g^3)$ operations in K.

2.6 Generic optimizations. We first summarize improvements for the generic arithmetic (see [12, 59, 77]).

Composition. The composition can be simplified by noting the recursive definition of the gcd of three polynomials and the fact that, most likely, a_1 and a_2 are coprime. This yields the first improvement. We are given two semi-reduced divisors $D_1 = \mathrm{div}(a_1, b_1)$, $D_2 = \mathrm{div}(a_2, b_2)$ defined over K, and want to compute a semi-reduced divisor $D = \mathrm{div}(a, b)$ defined over K such that $D \sim D_1 + D_2$ on $J_C(K)$.

1. If $a_1 = a_2$, then we put $d_1 := a_1$; $s_1 := 1$. Else use the half-extended Euclidean algorithm to compute $d_1, s_1 \in K[u]$ such that $d_1 = \gcd(a_1, a_2) = s_1a_1 + t_1a_2$ for some $t_1 \in K[u]$.

2. If $\deg d_1 = 0$, then we put $d := 1$; $x_1 := d_1^{-1} s_1$; $x_3 := 0$. Else use the extended Euclidean algorithm to compute $d, s_2, t_2 \in K[u]$ such that
$$d = \gcd(d_1, b_1 + b_2 + h) = s_2 d_1 + t_2(b_1 + b_2 + h) \ .$$
Then we compute $x_1 := s_1 s_2$ and put $x_3 := t_2$.
3. Then $D = \operatorname{div}(a, b)$ is given by the formulas
   (a) $a := a_1 a_2 / d^2$.
   (b) $b := b_1 + (x_1 a_1 (b_2 - b_1) + x_3 (f - b_1 h - b_1^2))/d \bmod a$.

Notice that we only require the half-extended Euclidean algorithm in the first step of the algorithm in case that $a_1 \neq a_2$. For random semi-reduced divisors, we expect that d_1 is a constant. Furthermore, we only need to compute x_1 and x_3 in order to determine the composition.

Cantor's reduction. Cantor suggested an improved reduction as follows. Given a semi-reduced divisor $D = \operatorname{div}(a, b)$ defined over K, the idea is to find a function $c - dv$ with nonzero $c, d \in K[u]$ such that the divisor $-((c - dv) - D)$ is reduced. The method applies if there exist $c, d, \lambda \in K[u]$ such that $c = \lambda a + db$, where $\deg c \leq (m + g)/2$, $\deg d \leq (m - g - 1)/2$, and $\gcd(\lambda, d) = 1$. Under these assumptions, we perform the following steps.

1. Use the extended Euclidean algorithm to find $c, d, \lambda \in K[u]$ with the above degree properties such that $c = \lambda a + db$.
2. Put $a_2 = \gcd(c, d) = \gcd(a, d)$, and compute $a_1 = a/a_2$, $c_1 = c/a_2$, $d_1 = d/a_2$, and $a_3 = (c_1^2 + c_1 d_1 h - d_1^2 f)/a_1$,
3. Use the half-extended Euclidean algorithm to find $d' \in K[u]$ such that $\gcd(d_1, a_3) = 1 = d_1 d' + e a_3$, i.e. $d_1 d' \equiv 1 \pmod{a_3}$, and compute $b_3 = -d' c_2 - h \bmod a_3$.
4. Finally, use the composition algorithm to compute the divisor sum D' of $\operatorname{div}(a_3, b_3)$ and $\operatorname{div}(a_2, b)$. Then D' is the reduced divisor equivalent to D.

The number of operations of Cantor's reduction is dominated by the complexity of the extended Euclidean algorithm, and requires $O(M(g) \log g)$ operations in K. Standard polynomial multiplication yields a complexity of $O(g^2 \log g)$. In fact, the reduction can be optimized so that one of the *gcd* computations can be avoided. If K has odd characteristic, the optimized reduction algorithm needs an expected number of $11g^2 + O(g)$ operations in K.

Improved reduction. We present an improved reduction method as introduced in [59, 77]. Given a semi-reduced divisor $D = \operatorname{div}(a, b)$ defined over K such that $\deg a > g$, we determine a reduced divisor $D' = \operatorname{div}(a', b')$ equivalent to D. Initially, we put $a_0 := a$ and $b_0 := b$.

1. Compute $a_1 := (f - hb_0 - b_0^2)/a_0$ and determine $q_1, b_1 \in K[u]$ such that
$$-h - b = q_1 a_1 + b_1 \quad , \quad \deg b_1 < \deg a_1 \ .$$
If $\operatorname{div}(a_1, b_1)$ is reduced, i.e. $\deg a_1 \leq g$, we put $i := 1$ and skip Step 2.
2. For $i \geq 2$, we compute $a_i, b_i, q_i \in K[u]$ such that
   (a) $a_i = a_{i-2} + q_{i-1}(b_{i-1} - b_{i-2})$
   (b) $-h - b_{i-1} = q_i a_i + b_i$, where $\deg b_i < \deg a_i$.

   We stop as soon as $\deg a_i \leq g$ for some $i \geq 2$.
3. We put $b' := b_i$ and normalize a_i to obtain a'.

If K has odd characteristic, this optimized algorithm computes the group law in an expected number of $17g^2 + O(g)$ operations in K. Doubling then needs an expected number of $16g^2 + O(g)$ operations. If K has characteristic 2, the group law needs an expected number of $14g^2 + O(g)$ operations in K. Doubling can be performed in an expected number of $11g^2 + O(g)$ operations.

2.7 Small genus. Recently there has been a surge in interest in the implementation of public-key cryptography using the jacobian of genus 2 and genus 3 hyperelliptic curves. Since the order of the jacobian of a genus g hyperelliptic curve over $\mathbb{F}_q$ is about q^g, one can achieve the same level of security with a genus 2 or genus 3 curve as with an elliptic curve, but by using a significantly smaller underlying field. For example, if an 80-bit security level is desired, then the underlying field should have order approximately 2^{160}, 2^{80} or 2^{54}, for $g = 1$, 2 or 3, respectively. [1] The smaller underlying fields may be advantageous in constrained environments with small processor architectures.

Several authors have presented explicit formulas for performing the group law in the jacobian. These formulas are specific to the genus 2 or genus 3 cases and improve upon Cantor's algorithm and its derivatives. A historical survey of the contributions can be found in [60]. Two recent contributions are those of Lange [47] and Pelzl et al. [60]. For genus 2 curves, Lange presented addition formulas that take 1 inversion, 22 multiplications, and 3 squarings in the underlying field, and doubling formulas that require 2 additional squarings. For genus 3 curves, Pelzl et al. gave addition formulas that take 1 inversion, 70 multiplications, and 6 squarings in the underlying field, and doubling formulas that take 1 inversion, 61 multiplications, and 10 squarings. Pelzl et al. also presented their implementation results which demonstrate that the performance of genus 2 and genus 3 curves can be quite competitive with elliptic curves. Finally, we mention that some work has been done on the hardware implementation of the arithmetic in genus 2 curves [10].

2.8 Hyperelliptic curves with efficiently computable endomorphisms. Let $k = \mathbb{F}_q$ be a finite field with q elements, and let $C : v^2 + h(u)v = f(u)$ be a non-singular hyperelliptic curve C of genus g in imaginary quadratic form. Furthermore, let $K = \mathbb{F}_{q^n}$ be an extension of k for $n \in \mathbb{N}$. C is called a *Koblitz curve* if we consider the jacobian $J_C(\mathbb{F}_{q^n})$, where n is at least 2. Note that the curve is defined over $\mathbb{F}_q$, whereas we consider the jacobian over the extension field $\mathbb{F}_{q^n}$. For practical applications, one considers curves defined over small finite fields $\mathbb{F}_q$ and then performs arithmetic in $\mathbb{F}_{q^n}$, where $n \gg q$. The main advantages are that (a) for small values of q, the cardinality of the Jacobian $J_C(\mathbb{F}_{q^n})$ can be easily computed for any $n \in \mathbb{N}$, and (b) computing m-folds of elements in $J_C(\mathbb{F}_{q^n})$ can be sped up considerably by making use of the *τ-adic expansion* of m, where τ denotes a formal root of the characteristic polynomial of the Frobenius. Since application of the Frobenius automorphism is relatively inexpensive (see Section 2.3), one is able to eliminate doubling of elements. Furthermore, the representations are shorter and more sparse.

These techniques were introduced by Koblitz [45] for elliptic curves, and improved by Solinas [75]. Günther, Lange and Stein [36] extended these methods to

[1] Recent progress by Thériault [81] in optimizing Gaudry's algorithm for computing logarithms in low genus hyperelliptic curves suggests that the underlying field for genus 3 curves may have to be larger than previously believed in order to obtain a desired security level.

the hyperelliptic curves $C_\alpha : v^2 + uv = u^5 + \alpha\, u^2 + 1$ ($\alpha = 0, 1$) which are defined over $\mathbb{F}_2$, and Lange [46] generalized them to all hyperelliptic Koblitz curves.

Another method for exploiting efficiently-computable endomorphisms to accelerate the computation of m-folds of elements in elliptic curves was presented by Gallant, Lambert and Vanstone [29], and generalized to hyperelliptic curves by Park, Jeong and Lim [56].

2.9 Hyperelliptic function fields and hyperelliptic curves. Let $k = \mathbb{F}_q$, and let C be an absolutely irreducible, hyperelliptic curve. In this section, we discuss how the arithmetic on the curve is related to the arithmetic in the function field $K(C)$ of C, where $k \le K \le \overline{k}$. We refer to Stichtenoth's book [79] for a discussion on algebraic function fields. In general, points on $C = C(\overline{k})$ are in one-to-one correspondence with prime divisors of the function field $\overline{k}(C)$. For $k \le K < \overline{k}$, prime divisors of $K(C)$ correspond to closed points of $C(K)$. The theory of divisors of hyperelliptic function fields is analogous to the theory of divisors of hyperelliptic curves. We therefore use the same notation.

A *hyperelliptic function field* L over K is a quadratic extension of the rational function field over K in one variable. A *prime divisor* P of L/K is the maximal ideal of a discrete valuation ring $\mathcal{O}$ of L with normalized discrete valuation v_P. (A prime divisor could also be defined as the equivalence class of a discrete valuation of L, i.e. a *place* of K.) The *degree* of P, denoted by $\deg P$, is the degree of the residue field $\mathcal{O}/P$. If K is algebraically closed, then $\deg P = 1$. A *divisor* of L/K is a finite formal sum of prime divisors $D = \sum_P m_P P$, where the $m_P \in \mathbb{Z}$ are 0 for all but finitely many P. We define the *degree* of D as $\deg D = \sum_P m_P \deg P$. The set $Div(L/K)$ of divisors of L/K, i.e. the free abelian group on the prime divisors of L/K, forms an abelian group. Let $Div^0(L/K)$ denote the group of degree 0 divisors of L/K. For $F \in L$, the *divisor of* F is $\mathrm{div}(F) = \sum_P v_P(F)P$. Now the set $P(L/K) = \{\mathrm{div}(F) : F \in L\}$ is a subgroup of $Div^0(L/K)$. The *jacobian* of L/K (or *divisor class group* of L/K) is defined by $J(L/K) = Div^0(L/K)/\mathcal{P}(L/K)$. We denote the class containing the divisor D by $[D]$. If D_1 and D_2 are degree 0 divisors in the same equivalence class of divisors in $J(L/K)$ we write $D_1 \sim D_2$. Again, $J(L/K)$ is a finite abelian group if K is a finite field. Its order $\mathfrak{h}$ is called the *divisor class number* of L/K.

Let $u \in L$ be a transcendental element such that $L/K(u)$ is a separable extension of degree 2. Let ∞ denote the infinite prime divisor of $K(u)$. If r denotes the number of extensions of ∞ in L, e its ramification index, f its degree, then we know that $ref = [L : k(u)] = 2$, since the extension is cyclic. Thus, we get three cases depending on whether (1) ∞ ramifies in L, i.e. $r = 1$, $e = 2$, $f = 1$; (2) ∞ splits in L, i.e. $r = 2$, $e = 1$, $f = 1$; or (3) ∞ is inert in L, i.e. $r = 1$, $e = 1$, $f = 2$. We ignore the third case, since it is degenerate. A constant field extension LK'/K', where K' is a quadratic extension of K, leads to the second case. We call a hyperelliptic function field of the form (1), i.e. in which the infinite prime divisor ∞ of $K(u)$ is ramified, an *imaginary quadratic function field.* A hyperelliptic function field in which the infinite prime divisor ∞ of $k(u)$ splits is called a *real quadratic function field.* In general, a non-singular, absolutely irreducible algebraic curve C is called a *hyperelliptic curve* over K if the corresponding function field $K(C)$ is hyperelliptic. C is called *imaginary quadratic* or *real quadratic*, respectively, depending on whether its function field is imaginary quadratic or real quadratic.

We will consider ideal arithmetic in hyperelliptic function fields. In the imaginary case, we will obtain nothing new. Let L/K be a hyperelliptic function field over K. Then the integral closure $\mathcal{O}$ of $K(u)$ in L is a Dedekind domain. The prime ideals are in one-to-one correspondence with finite prime divisors. If $I(\mathcal{O})$ denotes the group of fractional ideals of $\mathcal{O}$ and $P(\mathcal{O})$ the group of principal ideals of $\mathcal{O}$, then $P(\mathcal{O})$ is a subgroup of $I(\mathcal{O})$. The quotient group $I(\mathcal{O})/P(\mathcal{O})$ is called the *ideal class group* of L/K and its order $\mathfrak{h}'$ the *ideal class number* of L/K.

Imaginary quadratic curves. Let C be a (non-singular) hyperelliptic curve which is imaginary quadratic. Then C is of the form $C : v^2 + h(u)v = f(u)$, where $h, f \in k[u]$, f is monic, $\deg f = 2g + 1$, and $\deg h \leq g$. These are precisely the curves discussed in Section 2.1. Let $k \leq K \leq \overline{k}$ and $L = K(C)$. Here g denotes the genus of C which is equal to the genus of L. If the characteristic of the finite field $k = \mathbb{F}_q$ is different from 2, we can assume that $C : v^2 = f(u)$, where f is monic, $\deg f = 2g + 1$, and $L = K(u)(\sqrt{f(u)})$. Let $\mathcal{O}$ be the integral closure of $K(u)$ in L. Then, we know that the jacobian $J(L/K)$ is isomorphic to the ideal class group of L/K, and $\mathfrak{h} = \mathfrak{h}'$.

Any integral ideal $\mathfrak{a} \subseteq \mathcal{O}$ can be uniquely represented as $\mathfrak{a} = s(aK[u] + (b + v)K[u])$ with $s, a, b \in K[u]$, s, a monic, and a dividing $b^2 - bh - f$. Note that the *norm* of $\mathfrak{a}$ is then defined by $N(\mathfrak{a}) = as^2$. Thus, the degree of $\mathfrak{a}$ is given by $\deg \mathfrak{a} = \deg a + 2s$. If $s = 1$, then $\mathfrak{a}$ is called *primitive*. A primitive ideal $\mathfrak{a}$ is called *reduced* if $\deg \mathfrak{a} \leq g$. Summarizing, each reduced ideal $\mathfrak{a}$ has a unique representation of the form $\mathfrak{a} = aK[u] + (b+v)K[u]$, where $a, b \in K[u]$ with a monic, $\deg b < \deg a \leq g$, and $a|(b^2 - bh - f)$. Notice that (a, b) with these properties uniquely represent $\mathfrak{a}$.

It is well-known that each ideal class contains exactly one reduced ideal. Thus, the set of reduced ideals in $\mathcal{O}$ forms a set of representatives for the ideal class group. Furthermore, there is a canonical bijection between the jacobian $J(L/K)$ and the set of reduced ideals in $\mathcal{O}$. The group law $\mathfrak{a} * \mathfrak{b} = \mathfrak{c}$ on the set of reduced ideals is as follows:

1. **Composition**: Determine an ideal $\mathfrak{c}'$ representing the product of $\mathfrak{a}$ and $\mathfrak{b}$. Then $\mathfrak{a}\mathfrak{b} = \mathfrak{c}' = (s)\mathfrak{c}''$, where $\mathfrak{c}''$ is primitive.
2. **Reduction**: Compute the unique reduced ideal $\mathfrak{c}$ in the ideal class of $\mathfrak{a}\mathfrak{b}$.

The arithmetic in $J(L/K)$, and thus in the set of reduced ideals in $\mathcal{O}$, is the same as the arithmetic described in Sections 2.5 and 2.6 when changing the input of these algorithms slightly: replace (a, b) by $(a, -b)$, (a_1, b_1) by $(a_1, -b_1)$, and (a_2, b_2) by $(a_2, -b_2)$. [2]

Real quadratic curves. Let C be a (non-singular) hyperelliptic curve which is real quadratic. For $k = \mathbb{F}_q$, let $k \leq K \leq \overline{k}$ and $L = K(C)$. If the characteristic of k is different from 2, we can assume that $C : v^2 = f(u)$, where $f \in k[u]$ is monic, $\deg f = 2g + 2$, and $L = K(u)(\sqrt{f(u)})$. If the characteristic of k is 2, then C is of the form $C : v^2 + h(u)v = f(u)$, where $h, f \in k[u]$, h is monic, $\deg h = g + 1$, and either (a) $\deg f \leq 2g + 1$, or (b) $\deg f = 2g + 2$ and the leading coefficient of f is of the form $\beta^2 + \beta$ for some $\beta \in k$. Let $L = K(C)$ and let $\mathcal{O}$ be the integral closure of $K(u)$ in L. In both cases, g denotes the genus of C which is equal to the genus of L. Let ν_1 and ν_2 be the normalized valuations of L with respect to the two infinite prime divisors ∞_1 and ∞_2 of L. Any integral ideal $\mathfrak{a} \subseteq \mathcal{O}$ can be uniquely

[2] In our notation, a reduced ideal (a, b) corresponds to a reduced divisor $\mathrm{div}(a, -b)$.

represented as $\mathfrak{a} = s(aK[u] + (b+v)K[u])$ with $s, a, b \in K[u]$, s, a monic, and a dividing $b^2 - bh - f$. Then $N(\mathfrak{a}) = as^2$ is the *norm* of $\mathfrak{a}$ and $\deg \mathfrak{a} = \deg a + 2s$ is the degree of $\mathfrak{a}$. If $s = 1$, then $\mathfrak{a}$ is called *primitive*. A primitive ideal $\mathfrak{a}$ is called *reduced* if $\deg \mathfrak{a} \leq g$. Each reduced ideal $\mathfrak{a}$ has a unique representation $\mathfrak{a} = (a, b)$ such that $\mathfrak{a} = aK[u] + (b+v)K[u]$, where $a, b \in K[u]$ with a monic, $\deg b < \deg a \leq g$, and $a|(b^2 - bh - f)$.

Let $R \geq g+1$ denote the *regulator* of $\mathcal{O}$. Then $\mathfrak{h} = R\mathfrak{h}'$. If $\mathfrak{b}\mathfrak{a}^{-1} = \alpha\mathcal{O}$, we define the *distance* between $\mathfrak{a}$ and $\mathfrak{b}$ as $\delta(\mathfrak{b}, \mathfrak{a}) := \deg \alpha$, where $\deg(.)$ denotes the extension of the degree function from $K(u)$ to $K(C) \subseteq K((1/u))$; that is, $\deg(\alpha) = -\nu_1(\alpha)$. The distance is only well-defined and unique modulo R.

In the real quadratic case, each ideal class is represented by a whole cycle of reduced ideals. In general, we even expect to only have a few ideal classes except for some very special curves. The number of reduced ideals in each ideal class is bounded by R.

Infrastructure operation. We have the following *infrastructure operation* $\mathfrak{a} * \mathfrak{b} = \mathfrak{c}$ on the set of reduced ideals. Notice that the operation (computing the reduced product of two ideals) is in essence the same as in the imaginary case, but in the context of infrastructure, it does not induce a group on the set of reduced ideals (associativity fails).

1. **Composition**: Determine an ideal $\mathfrak{c}'$ representing the product of $\mathfrak{a}$ and $\mathfrak{b}$. Then $\mathfrak{a}\mathfrak{b} = \mathfrak{c}' = (s)\mathfrak{c}''$, where $\mathfrak{c}''$ is primitive.
2. **Reduction**: Compute a reduced ideal $\mathfrak{c}$ in the class of $\mathfrak{a}\mathfrak{b}$, i.e. reduce $\mathfrak{c}''$.

If both $\mathfrak{a}$ and $\mathfrak{b}$ are reduced, principal ideals, then we even have $\delta(\mathfrak{c}, \mathcal{O}) = \delta(\mathfrak{a}, \mathcal{O}) + \delta(\mathfrak{b}, \mathcal{O}) + \epsilon$, where $-2g \leq \epsilon \leq 0$. Hereby, $\delta(\mathfrak{c}, \mathcal{O})$ and ϵ can be computed with negligible extra effort. For details on the computation of the distance see [67, 58, 77].

The composition is identical to the composition in the imaginary case (see Sections 2.5 and 2.6) when replacing (a, b) by $(a, -b)$, (a_1, b_1) by $(a_1, -b_1)$, and (a_2, b_2) by $(a_2, -b_2)$, and noting the different conditions on the degrees of f and h. The reduction corresponds to the reduction of a real quadratic irrationality (see [6]). In a precomputation step, we compute the principal part $d = \lfloor V \rfloor$ of a root V of $v^2 + h(u)v - f(u) = 0$. The other root is $-V - h$. If $V = \sum_{i=-\infty}^{m} v_i u^i \in K((1/u))$, then $d = \sum_{i=0}^{m} v_i u^i$. Given $a, b \in K(u)$ representing a primitive ideal $\mathfrak{a}$ such that $\deg a > g$, we determine a reduced ideal $\mathfrak{a}' = (a', b')$ equivalent to $\mathfrak{a}$. Initially, we put $r_{-1} := 0$, $a_0 := a$ and $b_0 := b$.

1. Compute $r_0 \in K[u]$ such that $b_0 + d = q_0 a_0 + r_0$ and $\deg r_0 < \deg a_0$. Put $b_1 := d - r_0 + h = q_0 a_0 - b_0 + h$ and $a_1 := (f + hb_1 - b_1^2)/a_0$. If $\deg a_1 \leq g$, we put $i := 1$ and skip Step 2.
2. For $i \geq 2$, we compute $a_i, b_i, q_{i-1}, r_{i-1} \in K[u]$ such that
   (a) $b_{i-1} + d = q_{i-1} a_{i-1} + r_{i-1}$, where $\deg r_{i-1} < \deg a_{i-1}$.
   (b) $b_i := d - r_{i-1} + h = q_{i-1} a_{i-1} - b_{i-1} + h$.
   (c) $a_i := a_{i-2} + q_{i-1}(r_{i-1} - r_{i-2}) = (f + hb_{i+1} - b_{i+1}^2)/a_i$.
   We stop as soon as $\deg a_i \leq g$ for some $i \geq 2$.
3. We put $b' := b_i \bmod a_i$ and normalize a_i to obtain a'.

In order to keep track of the distance from $\mathfrak{a}$ to $\mathfrak{a}'$, we simply put $\delta_0 := 0$, and $\delta_i = \delta_{i-1} + \deg a_i - \deg(b_i - V - h)$ for $i \geq 1$. Then $\delta(\mathfrak{a}', \mathfrak{a}) = \delta_i$, where $i \geq 1$ is minimal with $\deg a_i \leq g$.

If K has odd characteristic, this algorithm computes the infrastructure operation in an expected number of $17g^2 + O(g)$ operations in K. Doubling needs an expected number of $16g^2 + O(g)$ operations. If K has even characteristic, the complexity has not been worked out explicitly. This is a topic for future work. We would expect a similar complexity as for the group operation in the imaginary quadratic case.

Baby steps. If one applies the above reduction formulas to a reduced ideal, the output will be another reduced ideal in the same ideal class. This operation is referred to as a *baby step*. The equivalent operation in the imaginary case is a trivial operation, since each ideal class contains a unique reduced ideal. Assume we perform l baby steps starting with the ideal $\mathfrak{a} = (a_0, b_0)$. If the characteristic of K is odd, then the first baby step i.e. the computation of (a_1, b_1), needs an expected number of $3g^2/2 + O(g)$ operations, whereas the computation of (a_i, b_i), $2 \le i \le l$, needs an expected number of $4g + O(1)$ operations in K. If K has even characteristic, we expect a similar complexity. This has not been worked out and it is a topic for future work.

Group operation. In [58], it was shown that there is a canonical bijection between the jacobian group $J(L/K)$ and the set of pairs $\{(\mathfrak{a}, n)\}$, where $\mathfrak{a}$ is a reduced ideal of $\mathcal{O}$ and n is an integer with $0 \le \deg(\mathfrak{a}) + n \le g$. For $\lambda \in \mathbb{R}$ we let $\delta(\mathfrak{b}, \mathfrak{a}, \lambda) := \max\{n = \delta(\mathfrak{b}, \mathfrak{a}) \pmod{R} \mid n \le \lambda\}$. Then, we have the following group law $(\mathfrak{a}, n_a) * (\mathfrak{b}, n_b) = (\mathfrak{d}, n)$ on the set of these pairs: Simply perform the infrastructure operation as above and then compute the ideal that is closest to $n_a + n_b + \deg s$.

1. **Composition**: Determine an ideal $\mathfrak{c}'$ representing the product of $\mathfrak{a}$ and $\mathfrak{b}$. Then $\mathfrak{a}\mathfrak{b} = \mathfrak{c}' = (s)\mathfrak{c}''$, where $\mathfrak{c}''$ is primitive.
2. **Reduction**: Compute a reduced ideal $\mathfrak{c}$ in the class of $\mathfrak{a}\mathfrak{b}$, i.e. reduce $\mathfrak{c}''$.
3. **Closest**: In the ideal class of $\mathfrak{a}\mathfrak{b}$, perform baby steps to determine a reduced ideal $\mathfrak{d}$ such that $\delta(\mathfrak{d}, \mathfrak{a}\mathfrak{b}, n_a + n_b)$ is maximal, and define $n = n_a + n_b - \delta(\mathfrak{d}, \mathfrak{a}\mathfrak{b}, n_a + n_b)$.

If the characteristic of K is odd, this algorithm computes the group law in an expected number of $19g^2 + O(g)$ operations in K. If the characteristic of K is even, the complexity of the group operation has not been worked out yet. This is a topic for future work.

Conversions. Finally, we want to discuss conversions from imaginary to real and from imaginary to real (see [58, 77, 16]). For genus 1, this was investigated in [76, 87].

Let C be an imaginary quadratic curve with function field $K(C)$. If the size of the finite field $k = \mathbb{F}_q$ satisfies $q \gg g^2$, then a birational transformation yields a real quadratic model of the curve. In most of the cryptographic applications, we may assume that $q \gg g^2$ is satisfied. For instance, if the characteristic of k is odd, then C admits a representation of the form $C : v^2 = f(u)$, where f is monic, $\deg f = 2g + 1$, and $K(C) = K(u)(\sqrt{f(u)})$. If $q \gg g^2$, then C can be represented as a real quadratic curve over the same finite field k by applying the following birational transformation. For a suitable $\beta \in k$, we let $x := 1/(u - \beta)$ and $D(x) := x^{2g+2} f(\beta + 1/x)$. Then, the curve $C' : y^2 = D(x)$ is real quadratic and $K(C) = K(C')$.

Conversely, let C be a real quadratic curve with function field $K(C)$. If there exists a ramified prime divisor of degree 1 in $K(C)$, then we are able to perform a birational transformation to an imaginary quadratic curve. If not, we have to perform a constant field extension of degree l, where l is the smallest degree of a ramified prime divisor. If K_l is an extension of K of degree l, then we work in the constant field extension $K(C)K_l$ over K_l. Notice that $l \leq 2g+2$. For instance, if the characteristic of k is odd, then C admits a representation of the form $C: y^2 = D(x)$, where D is monic, $\deg D = 2g+2$, and $K(C) = K(x)(\sqrt{D(x)})$. If $D(\alpha) = 0$ for some $\alpha \in K$, then $K(C)$ contains a ramified prime of degree 1, and we perform the following birational transformation. We let $u := 1/(x-\alpha)$ and $f(u) := u^{2g+2}D(\alpha + 1/u)$. Then, the curve $C' : v^2 = f(u)$ is imaginary quadratic and $K(C) = K(C')$. If K is algebraically closed, then such an α always exists. Summarizing, in cryptographic applications we may assume that a hyperelliptic curve admits a real quadratic and an imaginary quadratic representation up to a constant field extension.

If the characteristic of k is even, the conversions are slightly more technical. The idea and a description can be found in [16, Section 3.1.3].

2.10 Hyperelliptic NUCOMP. NUCOMP [70] is an algorithm invented by Daniel Shanks for composing and reducing positive definite binary quadratic forms. Cantor's algorithm is a generalization of the usual composition and reduction algorithms in this setting, and as shown in [41], the ideas in NUCOMP can be used for divisor addition.

When using any of the preceding variants of Cantor's algorithm, the component polynomials of the divisor produced after the composition step can have degree as large as $2g$, whereas the components of the reduced divisor have degree at most g. The main idea of NUCOMP is to keep the degrees of all intermediate operands small (degree close to g) by performing the reductions *before* completing the composition. Instead of computing the composite directly in Step 3 of the composition algorithm in Section 2.6, we first compute the rational continued fraction expansion of a_2/dU, where $U \equiv x_1(b_2 - b_1) + x_3(f - b_1h - b_1^2)/a_1 \pmod{a_2/d}$. This continued fraction expansion produces a sequence of partial quotients $q_0, q_1, \ldots, q_n$ with $q_i \in k[u]$, and as dU/a_2 is a close rational approximation of $(b+y)/a$ (recall a and b are the components of $\mathrm{div}(a,b) = \mathrm{div}(a_1,b_1) + \mathrm{div}(a_2,b_2)$), these partial quotients are almost identical to the sequence of q_i values that would be produced by reducing $\mathrm{div}(a,b)$. Given these q_i values, there exist formulas for computing a' and b' such that $\mathrm{div}(a',b') \sim \mathrm{div}(a,b)$ and $\mathrm{div}(a',b')$ is the divisor that would be produced by applying the reduction algorithm to $\mathrm{div}(a,b)$ but using the q_i values in place of those computed during the reduction process. Furthermore, it is possible to choose the number of these partial quotients to use so that the operands in the formulas have degree not much more than $g/2$ and that the components of $\mathrm{div}(a',b')$ have degree about g, i.e. $\mathrm{div}(a',b')$ is equivalent to $\mathrm{div}(a_1,b_1)+\mathrm{div}(a_2,b_2)$ and very close to being reduced (normally only 1 or 2 reduction steps away at most). For more details, see [41].

Computational results on applying NUCOMP to both imaginary and real hyperelliptic curve representations are presented in [41]. These results suggest that NUCOMP for arithmetic in both the jacobian and the infrastructure becomes superior to Cantor's algorithm for genus around 7, and also as the size of the finite field increases. However, the potential of NUCOMP has not been exploited fully — the first author has recently carried out computations which suggest that the trade-off point may be closer to genus 4. Although more investigation of NUCOMP is required (and is already underway), the results certainly indicate that NUCOMP is an efficient alternative to Cantor's algorithm for any applications of high-genus hyperelliptic curves, for example, the Weil descent attack on the ECDLP.

3 Index-calculus attacks on the HCDLP

Index-calculus algorithms have been successfully applied to several cryptographically interesting problems, including discrete logarithm computations in finite fields and class groups of number fields. In principle, the same ideas can be applied to discrete logarithm computations in the jacobian $J_C(k)$ of a hyperelliptic curve C of genus g over $k = \mathbb{F}_q$ in imaginary quadratic form. If the genus g is sufficiently large with respect to q, then a probabilistic algorithm with runtime subexponential in $n = q^g$ is known, namely $O(L_n[c])$ where

$$L_n[c] = \exp\left((c + o(1))\sqrt{\log n \log\log n}\right)$$

for some positive real constant c.

The notion of *smoothness* with respect to a set of small irreducible elements is fundamental to any index-calculus algorithm. In the context of hyperelliptic curves, the role of irreducibles is played by the *prime divisors*, those divisors $D = \operatorname{div}(a, b)$ for which a is an irreducible polynomial over k. A divisor D is said to be t-smooth if it decomposes as $D = \sum_{i=1}^{L} e_i \operatorname{div}(a_i, b_i)$ for $e_i \in \mathbb{Z}$ and $\max\{\deg a_i\} \leq t$. To determine the decomposition of $D = \operatorname{div}(a, b)$ into prime divisors, factor a into monic irreducibles over k as $a = a_1^{e_1} a_2^{e_2} \cdots a_L^{e_L}$ and let $b_i = b \bmod a_i$. Then $D = \sum_{i=1}^{L} e_i \operatorname{div}(a_i, b_i)$.

3.1 Main strategies. There are two main index-calculus variations that have been proposed for the HCDLP. In the following, $D_1 \in J_C(\mathbb{F}_q)$, $D_2 \in \langle D_1 \rangle$, and we wish to determine $l = \log_{D_1} D_2$.

Strategy 1. The first strategy uses the same basic idea as Hafner and McCurley's algorithm for discrete logarithm computation in imaginary quadratic number fields [37]. The first step is to compute the structure of $J_C(k)$ as a direct sum of cyclic subgroups. Then, representations of D_1 and D_2 on this direct sum are computed, after which solving the DLP reduces to the generalized Chinese remainder theorem.

The method proceeds as follows. Let $S = \{P_1, P_2, \ldots, P_n\}$ be the *factor base* consisting of all split and ramified prime divisors $P_i = \operatorname{div}(a_i, b_i)$ with $\deg a_i \leq t$ for some bound t. If the irreducible a_i splits, then only one of the two prime divisors over a_i, $\operatorname{div}(a_i, b_i)$ or $\operatorname{div}(a_i, -h - b_i)$, is included in S. The first stage of the algorithm consists of finding $m > n$ t-smooth principal divisors, each of which yields a *relation* of the form $\sum_j e_j P_j \sim 0$. If S generates $J_C(k)$, then the map $\phi : \mathbb{Z}^n \to J_C(k)$ where $\phi(e_1, e_2, \ldots, e_n) \mapsto \sum_j e_j P_j$ is a surjective homomorphism and $J_C(k) \cong \mathbb{Z}^n / \ker(\phi)$. Each relation yields an element $\vec{e_i} = (e_{i1}, e_{i2}, \ldots, e_{in}) \in \ker(\phi)$, and if the set of m relations forms a complete generating system of $\ker(\phi)$, then $J_C(k) \cong \mathbb{Z}/d_1\mathbb{Z} \oplus \cdots \oplus \mathbb{Z}/d_n\mathbb{Z}$ where $(d_1, d_2, \ldots, d_n)$ are the diagonal elements of the Smith normal form (SNF) of the *relation matrix* $A = (\vec{e}_1 \vec{e}_2 \ldots \vec{e}_m)$ (the relations are written as columns of A). Generators X_i of each cyclic subgroup $\mathbb{Z}/d_i\mathbb{Z}$ can be computed by finding the unimodular transformation matrices $P = (p_{ij})$ and $Q = (q_{ij})$ such that $P^{-1}AQ = \operatorname{SNF}(A)$ and computing $X_i = \sum_{j=1}^{n} p_{ji} P_j$.

The second stage of the algorithm consists of finding representations of D_1 and D_2 in $\mathbb{Z}/d_1\mathbb{Z} \oplus \cdots \oplus \mathbb{Z}/d_n\mathbb{Z}$. If D_1 and D_2 can be factored over S as $D_1 \sim \sum \alpha_i P_i$

and $D_2 \sim \sum \beta_i P_i$, then $D_1 = \sum \alpha'_i X_i$ and $D_2 \sim \sum \beta'_i X_i$ where $(\alpha'_1, \ldots, \alpha'_n) = P^{-1}(\alpha_1, \ldots, \alpha_n)^T$ and $(\beta'_1, \ldots, \beta'_n) = P^{-1}(\beta_1, \ldots, \beta_n)^T$. Finally, given the representations of $D_1 = (\alpha'_1, \alpha'_2, \ldots, \alpha'_n)$ and $D_2 = (\beta'_1, \beta'_2, \ldots, \beta'_n)$ in $\mathbb{Z}/d_1\mathbb{Z} \oplus \mathbb{Z}/d_2\mathbb{Z} \oplus \cdots \oplus \mathbb{Z}/d_n\mathbb{Z}$, the DLP can be solved by using the generalized Chinese remainder theorem to find $l \in \mathbb{Z}$ such that the congruences $\alpha'_i \equiv l\beta'_i \pmod{d_i}$, $1 \leq i \leq n$, are simultaneously satisfied.

Strategy 2. The second strategy improves on the first when $\#J_C(k)$ is known. As before, let $S = \{P_1, P_2, \ldots, P_n\}$ be a factor base consisting of all prime divisors of degree $\leq t$. Relations are found by attempting to factor divisors of the form $\alpha D_1 + \beta D_2$ over S. Every such t-smooth divisor yields a relation of the form $\alpha_i D_1 + \beta_i D_2 \sim R_i = \sum_j e_{ij} P_j$. When $n+1$ different relations have been found, linear algebra modulo $\#J_C(k)$ is applied to find a non-trivial linear combination $\sum_{i=1}^{n+1} \gamma_i \vec{e}_i = (0, 0, \ldots, 0)$, which implies that $\sum_{i=1}^{n+1} \gamma_i R_i = 0$. Thus, $\sum_{i=1}^{w+1} \gamma_i(\alpha_i D_1 + \beta_i D_2) = 0$, and $\log_{D_1} D_2 = -(\sum \gamma_i \alpha_i)/(\sum \gamma_i \beta_i) \bmod \#J_C(k)$. The assumption that $\#J_C(k)$ is known allows the relatively expensive Smith normal form computation of the previous strategy to be replaced by the solution of a linear system modulo $\#J_C(k)$.

Hybrid strategy. It should be noted that Vollmer has proposed another strategy in the context of quadratic number fields [84] that can easily be applied to the HCDLP. His strategy is a combination of the previous two, whereby the expensive Smith normal form computation is replaced by linear system solving over the integers, or, if the group order is known, modulo $\#J_C(k)$. First, n relations of the form $\sum_j e_{ij} P_j = 0$ are generated, followed by two extra relations of the form $-D_1 + \sum_j e_{(n+1)j} P_j = 0$ and $D_2 + \sum_j e_{(n+2)j} P_j = 0$. Form the augmented relation matrix

$$B = \begin{pmatrix} A & \vec{e}_{n+1} & \vec{e}_{n+2} \\ \vec{0} & 1 & 0 \\ \vec{0} & 0 & 1 \end{pmatrix}$$

where $A = (\vec{e}_1 \vec{e}_2 \ldots \vec{e}_n)$ is the matrix consisting of the first n relations written as columns. Then $D_1 = lD_2$ if and only if there exists a solution $\vec{x} \in \mathbb{Z}^{n+2}$ of $B\vec{x} = (0, \ldots, 0, 1, l)$, since this implies $-D_1 + lD_2 = 0$. To find l, write $B = \begin{pmatrix} B' \\ \vec{b} \end{pmatrix}$ and let $\vec{y} \in \mathbb{Z}^{n+2}$ be a solution to $B'\vec{y} = (0, \ldots, 0, 1)$. Then $l = \vec{b} \cdot \vec{y}$ is a solution to $D_1 = lD_2$ (not necessarily minimal). This method may be useful if a faster method exists to generate relations of the form $\sum_j e_j P_j = 0$ rather than $\sum_j e_j P_j = \alpha D_1 + \beta D_2$.

Analysis. In order for any of these strategies to yield an $O(L_{q^g}[c])$ subexponential algorithm, two conditions must be satisfied:

1. The prime divisors in S generate $J_C(k)$ and $n \in O(L_{q^g}[\rho])$ for some positive constant ρ. In other words, the size of the factor base can be taken to be subexponential in q^g.
2. (Smoothness assumption) The expected number of random divisors selected before a t-smooth divisor is found (assuming a factor base of size $O(L_{q^g}[\rho])$) is $O(L_{q^g}[\frac{a}{\rho}])$ for a positive constant $a < 1$.

The analysis consists of finding an optimal value of ρ such that the linear algebra and relation generating stages take the same amount of time. Under the first assumption, if $n \in O(L_{q^g}[\rho])$, then the linear algebra step requires time $O(L_{q^g}[l\rho])$ for some constant l, where the required linear algebra can be performed in time

$O(n^l)$ for $n \times n$ matrices. The expected runtime for relation generation depends on the smoothness assumption, the time required to generate and test a single divisor for smoothness, and the expected number of smooth divisors required before the discrete logarithm is found.

In all three of the above methods, a computed discrete logarithm is always correct. In order to verify the insolubility of a DLP instance, it must be verified that the set of relations generates $\ker(\phi)$. This can be accomplished by checking that h', the determinant of the lattice generated by the relations, is equal to $\#J_C(k)$, or if $\#J_C(k)$ is unknown, checking that $H^* < h' < 2H^*$ where $H^* < \#J_C(k) < 2H^*$. Such an approximation H^* to $\#J_C(k)$ can be computed efficiently using the methods described in [78].

Note that any algorithms following the three strategies above are probabilistic of Las Vegas type. That is, if the algorithm terminates the answer is guaranteed to be correct, but the runtime given is the *expected* time the algorithm takes to terminate.

As with Pollard's rho method (see [28, 86]), the index-calculus methods described above can be improved in practice if the hyperelliptic curve has a non-trivial automorphism that is efficiently computable. Suppose σ is an automorphism of order m on $J_C(k)$. In this case, Gaudry [31] observed that the factor base size can be reduced by a factor of m by including only one prime divisor out of m in each orbit under σ. Let G_i denote these representatives. If $R = P_1 + P_2 + \cdots + P_r$ is the decomposition of R into prime divisors of degree $\leq t$, then we can write $R = \theta^{l_1} G_1 + \ldots \theta^{l_r} G_r$, $0 \leq l_i < m$, because there exists an integer θ such that $\sigma(D) = \theta D$ for any divisor D and P_i lies in the orbit of G_i under σ. This technique will yield an improved runtime, as fewer relations are required and the relation matrix will have smaller dimension.

3.2 Proposed algorithms. We now describe the algorithms presented in the literature. Their main differences lie in the means by which relations are generated, the constant c obtained in the runtime analysis, and the number of heuristic assumptions required to obtain the claimed expected runtime.

The ADH algorithm. Adleman, DeMarrais, and Huang (ADH) [3] presented the first index-calculus algorithm for solving the HCDLP. Their algorithm was described for the case q an odd prime, and was later extended to arbitrary q by Bauer [8]. The ADH algorithm does not assume that $\#J_C(k)$ is known and follows Strategy 1. Let C be defined by $v^2 + hv = f$ over k. Relations are found by attempting to factor random semi-reduced principal divisors generated by rational functions of the form $Av + B$, $A, B \in k[u]$. Every such smooth divisor yields a relation of the form $(A_i v + B_i) = \sum_j e_{ij} P_j \sim 0$. Thus, the relation generation stage of the algorithm proceeds by randomly choosing $A, B \in k[u]$ and attempting to factor $(Av + B)$ over the factor base S until sufficiently many relations are found that the rank of the relation matrix is n and the discrete logarithm can successfully be computed as in Strategy 1.

Factoring $(Av + B)$ can be done by factoring its norm $B^2 + ABh - A^2 f = \prod_{j=1}^{n} a_j^{e'_j}$ where $P_j = \mathrm{div}(a_j, b_j)$ are the prime divisors in the factor base. Set $e_j = e'_j$ if $Ab_j + B \equiv 0 \pmod{a_j}$, otherwise set $e_j = -e'_j$. Then $(Av + B) = \sum_j e_j P_j$ is the decomposition of the principal divisor $(Av + B)$.

If $\log q \leq (2g+1)^{0.98}$, the ADH algorithm can be shown to run in expected time $O(L_{q^{2g+1}}[c])$ for some positive real constant $c < 2.313$. Note that the bound on the constant c is the corrected value due to Bauer [8]. However, the analysis in [3] and [8] is not rigorous. In particular, the following two (unproved) assumptions are made:

1. $J_C(k)$ is generated by all prime divisors of degree at most $\log_q L_{q^{2g+1}}[c]$ for some positive real constant c.
2. Let $D = \operatorname{div}(a, b)$ be a random divisor. The probability that D is t-smooth is the same as the probability that a random polynomial of $\deg a$ over k decomposes into factors of degree at most t. Bauer [8] shows that this implies that the expected number of trials before a relation is found is $O(L_{q^{2g+1}}[9/(16c)])$ provided that $\log q \leq (2g+1)^{0.98}$.

Under these assumptions, the dominant step of the algorithm is the last one in which relations are generated until the relation matrix has full rank and the discrete logarithm is successfully computed. In each iteration of this step, the linear algebra takes time $O(L_{q^{2g+1}}[lc])$ and the relation generation takes time $O(L_{q^{2g+1}}[9/(16c)])$, yielding an optimal value of $c = 3/(4\sqrt{l})$. The expected number of repetitions of this step is $O(L_{q^{2g+1}}[c])$, yielding an overall expected runtime of $O(L_{q^{2g+1}}[c])$ with $c = 3(l+1)/(4\sqrt{l})$. The analysis in [3] assumes $l < 7.376$ for the linear algebra, yielding $c < 2.313$.

The MST algorithm. The second index-calculus algorithm for solving the HCDLP was proposed by Müller, Stein, and Thiel [53]. Their algorithm, which works for any odd-characteristic finite field, solves the discrete logarithm problem in the infrastructure of a real quadratic function field. For elliptic curves, it was shown in [76] (odd characteristic) and [87] (even characteristic) that the ECDLP is polynomially equivalent to the infrastructure discrete logarithm problem as defined below. Paulus and Rück [58] generalized this result and showed that the HCDLP is equivalent to the infrastructure discrete logarithm problem up to a constant field extension (see Section 2.9 for conversions).

Let $k = \mathbb{F}_q$ be a finite field of odd characteristic, and let $C : v^2 = f$, where $f \in k[u]$ is monic, $\deg f = 2g+2$. Then $L = k(C) = k(u)(\sqrt{f})$ is a real quadratic function field. The *infrastructure discrete logarithm problem* (IDLP) is defined as follows: given polynomials $a, b \in k[u]$ representing a reduced, principal ideal $\mathfrak{a} = aK[u] + (b+v)K[u]$; find its distance $\delta(\mathfrak{a}, \mathcal{O})$. That is, find $\deg(\alpha)$, where $\mathfrak{a} = (\alpha)$. Notice that there exists a unique (up to constants) generator for $\mathfrak{a}$ such that $0 \leq \deg(\alpha) < R$. [3]

The algorithm basically follows Strategy 1 from above, and is based on similar ideas of Buchmann [11] and Abel [1] in real quadratic number fields. First, let $S = \{\mathfrak{p}_1, \mathfrak{p}_2, \ldots, \mathfrak{p}_n\}$ be the factor base consisting of all split and ramified prime ideals P_i with $\deg P_i \leq t$ for some bound t. The first stage of the algorithm consists of finding $m > n$ t-smooth principal ideals, each of which yields a relation of the form $\mathfrak{b} = \prod_j \mathfrak{p}_j^{e_j} = (\beta)$. Each relation yields an element $\vec{v}_j = (e_{j1}, e_{j2}, \ldots, e_{jn}, \deg(\beta_j)) \in \mathbb{Z}^{n+1}$. Let Γ_t be the set of these elements $\vec{v}_j \in \mathbb{Z}^{n+1}$, and let Γ'_t be the set of elements $\vec{e}_j = (e_{j1}, e_{j2}, \ldots, e_{jn}) \in \mathbb{Z}^n$ such that $\prod_j \mathfrak{p}_j^{e_j}$ is principal. If the set S generates the

[3] Note that we could define the IDLP more general as follows: given two reduced, equivalent ideals $\mathfrak{a} = (a, b)$ and $\mathfrak{b} = (a', b')$; find $\delta(\mathfrak{b}, \mathfrak{a})$. Multiplication of the equation $\mathfrak{b} = (\alpha)\mathfrak{a}$ with the ideal $\overline{\mathfrak{a}} = (a, -b-v)$ and reduction of $\mathfrak{b}\overline{\mathfrak{a}}$ yields the IDLP in the principal ideal class, since $\mathfrak{a}\overline{\mathfrak{a}} = (a)$.

ideal class group, then Γ_t is a $(n+1)$-dimensional lattice of determinant $\mathfrak{h} = \mathfrak{h}'R$, and Γ_t' is a n-dimensional lattice of determinant $\mathfrak{h}'$. The following process is used to generate relations:

1. Randomly select $(e_1, e_2, \ldots, e_n) \in \{0, \ldots, q^{2g+2}\}^n$ and compute $\mathfrak{c} = \prod_j \mathfrak{p}_j^{e_j}$,
2. Find a reduced ideal $\mathfrak{b} \sim \mathfrak{c}$ and $\deg(\beta)$, where $\beta \in L$ is such that $\mathfrak{b} = (\beta)\mathfrak{c}$,
3. If $\mathfrak{b}$ can be factored over the factor base S, $\mathfrak{b} = \prod_j \mathfrak{p}_j^{v_j}$, then $\mathfrak{b}\mathfrak{c}^{-1} = \prod_j \mathfrak{p}_j^{(v_j - e_j)} = (\beta)$ yields a relation.

In fact, we are neglecting an additional distance condition which is mainly of theoretical interest.

Assume we have found a complete generating system $\{\vec{v}_1, \ldots, \vec{v}_m\}$ of Γ_t. When removing the last coordinate in each $\vec{v}_j$ we obtain a complete generating system $\{\vec{v}_1', \ldots, \vec{v}_m'\}$ of Γ_t'. Let $A = (\vec{v}_1 \vec{v}_2 \ldots \vec{v}_m)$ (the vectors are written as columns of A) and, similarly, let $A' = (\vec{v}_1' \vec{v}_2' \ldots \vec{v}_m')$. In order to solve the IDLP for some reduced, principal ideal $\mathfrak{a}$, we proceed as follows. If $\mathfrak{a}$ can be factored over S as $\mathfrak{a} = \prod_j \mathfrak{p}_j^{a_j}$, then $\vec{a} = (a_1, a_2, \ldots, a_n) \in \Gamma_t'$, since $\mathfrak{a}$ is principal. Hence, there exists a solution $\vec{x} = (x_1, x_2, \ldots, x_m) \in \mathbb{Z}^m$ to the linear system of equations $A'\vec{x} = \vec{a}$. Thus, $\alpha = \prod_j \beta_j^{x_j}$ is a generator of $\mathfrak{a}$, and $\deg(\alpha) = \sum_j x_j \deg(\beta_j)$. If $\mathfrak{a}$ does not factor over the factor base, we perform baby steps to find a reduced, principal ideal $\mathfrak{b} \sim \mathfrak{a}$ that factors. In order to compute $\deg(\alpha) \pmod{R}$, we need to know R in addition. This can be accomplished by first determining $\det \Gamma_t$ as well as $\det \Gamma_t'$, and then computing $R = \det \Gamma_t / \det \Gamma_t'$.

Unlike the ADH algorithm, the analysis of the MST algorithm does not depend on any unproven assumptions. The authors prove that there exists a generating set of the ideal class group whose cardinality is polynomial in q^g, in particular that the set of prime ideals of degree at most $\lceil 2 \log_q (4g-2) \rceil$ is a generating set. In addition, an explicit estimate of the probability that a random reduced divisor is t smooth is derived, so the smoothness assumption used to analyze the ADH algorithm is not required. The second of these results is described in detail by Enge and Stein [20]. Finally, the analysis of the algorithm is not only rigorous, the complexity is better than the complexity of the ADH method. In [53], it is shown that that if $2g + 2 \geq \log q$, then with $n \in O(L_{q^{2g+2}}[\rho])$. the expected number of trials before finding a relation is $O(L_{q^{2g+2}}[\frac{1}{4\rho}])$. The linear algebra step was assumed to take time $O(L_{q^{2g+2}}[5\rho])$ and the relation generation stage takes time $O(L_{q^{2g+2}}[2\rho + \frac{1}{4\rho}])$, as $m \in O(L_{q^{2g+2}}[\rho])$ relations are required and it takes time $O(L_{q^{2g+2}}[\rho])$ to evaluate a random power-product of ideals. This yields an optimal value for ρ of $\rho = 5/2\sqrt{3}$ and an overall expected runtime of $O(L_{q^{2g+2}}[1.44])$.

Flassenberg and Paulus' sieving algorithm. Flassenberg and Paulus [21] were the first to apply Hafner and McCurley's relation generation strategy from [37] to the HCDLP. Their methods work for curves defined over any odd-characteristic finite field. The general method used follows Strategy 1 from above, and the following process is used to generate relations:

1. Randomly select $(e_1, e_2, \ldots, e_n) \in \mathbb{Z}^n$ and compute $D = \sum_j e_j P_j$,
2. Find a reduced divisor $D' \sim D$,
3. If D' is t-smooth with $D' = \sum_j v_j P_j$, then $D' - D = \sum_j (v_j - e_j) P_j \sim 0$ yields a relation.

The most intriguing aspect of Flassenberg and Paulus' contribution is their description of a sieving technique to potentially improve this method in practice. Sieving is a powerful method that lies at the heart of the best factoring and finite field discrete logarithm algorithms. It is based on the simple observation that for any polynomial $f(X)$ with coefficients in a ring R, if $p, x_0 \in R$ and $p \mid f(x_0)$, then $p \mid f(x_0 + ip)$ for all $i \in R$. In the case $R = \mathbb{Z}$, all values of $x \in [-M, M]$ for which $f(x)$ is smooth can be identified in roughly the same amount of time required to test a single value of $f(x)$ using trial division using an analogue of the sieve of Eratosthenes.

Sieving can be used in the HCDLP context because finding *any* t-smooth divisor $D' \sim D$ yields a relation. Consider the norm form $f(X, Y) \in (k[u])[X, Y]$ of a divisor $D = \operatorname{div}(a, b)$. Flassenberg and Paulus show that for any $x, y \in k[u]$ there exists a divisor $D' = \operatorname{div}(f(x, y), b') \sim D$. Thus, we can sieve the polynomial $f(X) = f(X, 1)$ (or $f(X, Y)$, as described in [21] to find x such that $f(x)$ is t-smooth—each such x yields a relation.

Unfortunately, sieving polynomials in $(k[u])[X]$ is not as easy as polynomials in $\mathbb{Z}[X]$. In the latter case, an integer array A is set up whose cells correspond to integers in the interval $[-M, M]$. Each cell in the array is initialized to zero, and $\log p$ is added to each $A[x]$ if $p \mid f(x)$. Once the smallest $x_0 \in [-M, M]$ for which $p \mid f(x_0)$ is found (by solving $f(X) \equiv 0 \pmod p$), all remaining values of x for which $p \mid f(x)$ can be found by jumping through A in steps of size p. To generalize this approach to polynomials over $(k[u])[X]$, one first has to decide how to set up the sieving array such that the indices correspond to polynomials in $k[u]$. Secondly, given $x_0 \in k[u]$ such that $p \mid f(x_0)$, one has to find an efficient procedure to find the other $x = x_0 + ip$, $i \in k[u]$, such that $p \mid f(x)$. Flassenberg and Paulus solve these problems by identifying each polynomial $f \in k[u]$ with $f(q)$ when $k = \mathbb{F}_q$. This solves the problem of creating the sieve array, but the procedure of finding all x for which $p \mid f(x)$ is more complicated than the integer polynomial case because the jumps in the array from one x value to the next are not the same size, necessitating extra bookkeeping.

No analysis of the basic algorithm nor the sieve-based algorithm is provided in [21], but some computational evidence that both are superior to the ADH strategy [3] is provided. We provide a summary of these computations in Section 5.

Bauer and Enge's algorithms for arbitrary finite fields. The ADH algorithm was independently generalized to hyperelliptic curves over arbitrary finite fields by Bauer [8] and Enge [17]. Bauer's algorithm has the same form as the ADH algorithm, and is shown to have the same expected run-time under the same heuristic assumptions.

Like Flassenberg and Paulus [21], Enge's algorithm [17, 18] applies the method of Hafner and McCurley [37] to the HCDLP. The same relation generation strategy is used without sieving. In contrast to Bauer's algorithm, Enge does not use any heuristic assumptions in his analysis — the results of [53] are used to prove that the factor base generates $J_C(k)$ and those of [20] to justify the smoothness assumption. In addition, Enge was the first to illustrate the dependence of the runtime on the ratio between g and $\log q$. He proves that if $g \geq \vartheta \log q$ for a positive constant ϑ, then $n \in O(L_{q^g}[\rho + \frac{1}{\sqrt{\vartheta}}])$ and the expected number of trials before finding a relation is $O(L_{q^g}[\frac{1}{2\rho}])$. The linear algebra step takes time $O(L_{q^g}[l\rho])$ and the relation generation stage takes time $O(L_{q^g}[2\rho + \frac{2}{\sqrt{\vartheta}} + \frac{1}{2\rho}])$, as n relations are required and it takes time $O(n^{1+o(1)})$ to generate one random linear combination of factor base elements. Taking $l = 4$ for the linear algebra we obtain an optimal value of $\rho = \frac{1}{2}\left(\sqrt{1+\frac{1}{\vartheta}} - \sqrt{\frac{1}{\vartheta}}\right)$ and an overall expected runtime of $O(L_{q^g}[c])$ with

$$c = 2\left(\sqrt{1+\frac{1}{\vartheta}} + \sqrt{\frac{1}{\vartheta}}\right) \quad .$$

Note that this runtime is that derived in [18], in which a generic version of this strategy which yields a subexponential discrete logarithm algorithm in any setting for which the corresponding smoothness assumption holds is described. As Enge states in [17], this algorithm is designed with a simplified analysis in mind rather than practical performance, and hence no computational data is provided.

Gaudry's algorithm for small genus. None of the preceding algorithms are specifically designed to take advantage of the cryptographically interesting situation where $\#J_C(k)$ is known. They can use this extra information by performing the required linear algebra modulo $\#J_C(k)$, but Gaudry [31] was the first to present an index-calculus algorithm designed with this in mind. His algorithm follows the outline described above as Strategy 2. After the factor base S is constructed (only degree 1 prime divisors are included), a random walk á la Teske [80] is performed in the set of reduced divisors equivalent to $\alpha D_1 + \beta D_2$. Each 1-smooth divisor encountered yields a relation, and the remainder of the algorithm proceeds as described above.

In addition to faster linear algebra modulo $\#J_C(k)$, the process of generating relations is quite efficient—each new candidate generated via the random walk requires only one addition in $J_C(k)$. In order to determine whether a given divisor $\mathrm{div}(a, b)$ is 1-smooth, it suffices to check whether the degree of $\gcd(u^q - u, a)$ is the same as $\deg a$ (provided that a has no repeated factors), as $\gcd(u^q - u, a)$ is the product of the distinct degree 1 irreducible polynomials that divide a. In practice, the possibility that a has repeated factors can either be ignored, or detected by computing the degree of $\gcd((u^q - u)^g, a)$.

Perhaps the most important contribution in [31] is the analysis of the algorithm. The analyses of the preceding algorithms suggest that they are only applicable when the genus is large with respect to q. In order to gain insight into the small genus case, Gaudry analyzed his algorithm assuming fixed genus and letting q vary. He was able to show that his algorithm runs in expected time $O(g^3q^2\log^2 q + g^2g!q\log^2 q)$ for fixed genus. Since generic DLP algorithms run in time $O(q^2)$ for fixed genus, Gaudry's result indicates that the HCDLP can be solved more efficiently than the generic algorithms if $g > 4$. Thus, Gaudry's analysis yielded the most precise statement as to which genera admit more efficient DLP algorithms than the worst-case generic algorithms. This analysis is backed up by some computations, which will be described in Section 5.

Enge and Gaudry [19] also describe a variation of this approach suitable for large genus hyperelliptic curves. The algorithm described is essentially a generic version of Strategy 2 above, suitable for any setting in which the group order is known and the requisite smoothness assumption holds. As in Enge's algorithm [18], the factor base contains all prime divisors of degree $\leq t$, and the smoothness testing idea is extended to check whether $a = \mathrm{lcm}\left(\{\gcd(u^{q^i} - u, a) \mid 1 \leq i \leq t\}\right)$ (testing for repeated factors of a can be handled as above). Using the smoothness results proved in [20], an expected runtime of $O(L_{q^g}[c])$ with

$$c = \sqrt{2}\left(\sqrt{1+\frac{1}{2\vartheta}}+\sqrt{\frac{1}{2\vartheta}}\right)$$

is proved provided that $g \geq \vartheta \log q$ for some positive constant ϑ. The improvement over the running time proved in [18] follows from the fact that faster linear algebra methods, randomized Lanczos for linear system solving versus Smith normal form computation of an integer matrix, are employed when the group order is known. Thus, a value of $l = 2$ rather than 4 can be used in the analysis, yielding the claimed expected runtime.

Thériault's Algorithm. Very recently, Thériault [81] has described an improved version of Gaudry's algorithm for small genus. The first improvement is to use only a fraction of the degree 1 prime divisors in the factor base. Although this reduces the probability of finding smooth divisors, the cost of linear algebra is reduced because the resulting relation matrix has smaller dimension. By balancing the relation generation and linear algebra stages, Thériault finds the optimal fraction of degree 1 prime divisors to use and obtains a runtime of $O(g^5 q^{2-\frac{2}{g+1}+\varepsilon})$.

A second variation of this algorithm also incorporates the large prime strategy, a well-known technique from other index-calculus algorithms. The idea is to keep track of divisors which factor completely over the factor base except for one additional "large prime" divisor. If two such partial relations are found with either the same large prime factor or large prime factors which are negations of each other, they can be combined to form a relation. By incorporating the large prime variant into the previous algorithm, Thériault obtains a runtime of $O(g^5 q^{2-\frac{4}{2g+1}+\varepsilon})$.

Notice that both algorithms are asymptotically faster than Gaudry's algorithm provided that $q > (g-1)!$ in the case of the first algorithm and $q > (g-1)!/g$ for the second. More importantly, both algorithms are asymptotically faster than generic algorithms for $g \geq 3$, suggesting that genus 3 hyperelliptic curves may not be as secure as previously believed.

4 Weil descent attack on the ECDLP

Frey [23] first proposed using Weil descent as a means to reduce the ECDLP in elliptic curves over finite fields $\mathbb{F}_{q^n}$ to the DLP in the jacobian variety of an algebraic curve of large genus defined over a proper subfield $\mathbb{F}_q$ of $\mathbb{F}_{q^n}$. Let l and n be positive integers. Let $q = 2^l$, and let $k = \mathbb{F}_q$ and $K = \mathbb{F}_{q^n}$. Consider the non-supersingular elliptic curve E defined over K by the equation

$$E\ :\ y^2 + xy = x^3 + ax^2 + b, \quad a \in K,\ b \in K^*.$$

We assume that $\#E(K) = dr$ where d is small (e.g., $d = 2$ or $d = 4$) and r is prime. Hence $r \approx q^n$. Let $b_i = b^{q^i}$, and define

$$m(b) = \dim_{\mathbb{F}_2}(\mathrm{Span}_{\mathbb{F}_2}\{(1, b_0^{1/2}), (1, b_1^{1/2}), \ldots, (1, b_{n-1}^{1/2})\}). \tag{4.1}$$

Assume now that either n is odd, or $m(b) = n$, or $\mathrm{Tr}_{K/\mathbb{F}_2}(a) = 0$. Gaudry, Hess and Smart [34] (see also [26, 38]) showed how Weil descent can be used to reduce the ECDLP problem in the subgroup of order r of $E(K)$ to the discrete logarithm problem in a subgroup of order r of the jacobian $J_C(k)$ of a hyperelliptic curve C of genus g defined over k. One first constructs the Weil restriction $W_{E/k}$ of scalars of E, which is an n-dimensional abelian variety over k. Then, $W_{E/k}$ is intersected with $n-1$ hyperplanes to eventually obtain the hyperelliptic curve C. The reduction algorithm together with the best algorithm available for solving the resulting HCDLP instance is called the *GHS attack* on the ECDLP. The genus of C is $g = 2^{m-1}$ or $2^{m-1}-1$, where $m = m(b)$.

In order for the GHS attack to be considered successful in attacking the ECDLP in $E(K)$, the DLP in $J_C(k)$ should be solvable in less time than it takes to solve the ECDLP instance using Pollard's rho algorithm. In general, $m \approx n$ whence $g \approx 2^{n-1}$ and $\#J_C(k) \approx q^{2^{n-1}}$ and the GHS attack fails. The GHS attack will only succeed if m is small, say $m \approx \log_2 n$, because then $g \approx n$ and $\#J_C(k) \approx q^n$.

The formula (4.1) was analyzed in [49], and it was shown that the GHS attack fails for all elliptic curves over fields $\mathbb{F}_{2^n}$ where $n \in [160, 600]$ is prime. However, the GHS attack has been shown to be successful for some elliptic curves over finite fields $\mathbb{F}_{2^n}$ where n is composite. In [40], an instance of the ECDLP in an elliptic curve over $\mathbb{F}_{2^{124}}$ was solved by first reducing it to an instance of the HCDLP in a hyperelliptic curve of genus 31 over $\mathbb{F}_{2^4}$, and then solving the latter using the Enge-Gaudry algorithm. This is noteworthy because Pollard's rho algorithm for solving the ECDLP instance is infeasible, while the HCDLP instance was solved in a few days on a small network of workstations. The effectiveness of the GHS attack for composite $n \in [100, 600]$ was further analyzed in [48], where the elliptic curves most susceptible to the GHS attack were identified and enumerated. Because of this relationship between the ECDLP and the HCDLP, improvements, both concrete and asymptotic, in algorithms for solving the HCDLP can increase the vulnerability of elliptic curves to the GHS attack.

Some fields were shown in [50] to be *weak* for elliptic curve cryptography in the sense that the GHS attack can be used to solve the ECDLP faster than Pollard's rho algorithm for all cryptographically interesting elliptic curves over these fields. An example of a weak field is $\mathbb{F}_{2^{210}}$ since the GHS attack can be used to reduce the ECDLP in all cryptographically interesting elliptic curves over $\mathbb{F}_{2^{210}}$ to an instance of the HCDLP for a genus 15 or 16 hyperelliptic curve over $\mathbb{F}_{2^{42}}$. While the Enge-Gaudry for solving the HCDLP instance is significantly faster than Pollard's rho algorithm for the ECDLP in $E(\mathbb{F}_{2^{210}})$, it is still infeasible with today's computer resources. However, if significant advances are made for the HCDLP for genus 15 and 16 curves over $\mathbb{F}_{2^{42}}$, then the finite field $\mathbb{F}_{2^{210}}$ may be shown to be *bad* for elliptic curve cryptography—that is, the ECDLP would be feasible for all elliptic curves over $\mathbb{F}_{2^{210}}$.

5 Implementations of index-calculus algorithms

As most of the interest in the HCDLP is confined to the low genus case (especially elliptic curves), few computational results using index-calculus algorithms have been published. The first implementation was reported by Paulus [57] (and later extended by Flassenberg and Paulus [21]), who described the computation of the structure of $J_C(k)$ for various curves of the form $v^2 = u^{2g+1} + 2u + 1$ over prime fields. The genus ranges from 1 to 12 and the prime fields considered are $\mathbb{F}_{11}$, $\mathbb{F}_{101}$, $\mathbb{F}_{1009}$, $\mathbb{F}_{10007}$, $\mathbb{F}_{100003}$. Runtimes are presented for a version of the algorithm without sieving, a version using only trial division (á la Hafner and McCurley), a sieving version, and a baby-step giant-step implementation. The runtimes indicate that their sieving strategy is the fastest of the three index-calculus variants, and becomes superior to baby-step giant-step as the genus grows. Although discrete logarithms were not computed, the runtimes would remain approximately the same, as determining the structure of $J_C(k)$ is the most time-consuming part of the algorithm.

Smart [73] described an implementation of a variation of the ADH algorithm [3] that incorporates the strategy of Flassenberg and Paulus. A sieving strategy analogous to the number field sieve factoring algorithm is employed to simultaneously test several principal divisors of the form $(Av + B)$ for smoothness, rather than testing them one at a time using trial division. Only a subset of the examples in [21] are computed, and the runtimes observed provide further evidence that the ADH algorithm, even with sieving, is inefficient in practice.

Gaudry [31] presented the first computations of discrete logarithms on curves with known large jacobians. Two examples were presented, the first is a genus 6 curve over $\mathbb{F}_{5026243}$ and the second a genus 6 curve over $\mathbb{F}_{2^{23}}$. Both examples have $\#J_C(k)$ on the order of 10^{40}, but also have non-trivial automorphisms which were used to accelerate the algorithm. These computations are significant in that the group size is larger than what can be handled using the Pollard's rho method.

Gaudry, Hess, and Smart [34] describe an application of Gaudry's algorithm [31] to the computation of discrete logarithms on a genus 4 curve defined over $\mathbb{F}_{2^{21}}$. The algorithm is modified to use a reduced factor base by including only degree one prime divisors of norm $v + \alpha$ where the three low-order bits of α are zero. In addition to reducing the number of relations required and the size of the relation matrix, many smoothness candidates $\operatorname{div}(a, b)$ can be eliminated quickly by checking whether the three low-order bits in the constant term of a are zero. The entire computation took about 25% less time than Pollard's rho method would have taken, thus providing further evidence that the HCDLP for genus 4 can be solved faster with index-calculus methods than with generic methods.

Jacobson, Menezes, and Stein [40] implemented an improved version of the Enge-Gaudry algorithm [19] to solve HCDLP instances on curves defined over characteristic 2 finite fields that arise as a result of the Weil Descent attack on elliptic curves. The smoothness-testing strategy was optimized further, and an analysis of the optimal choice of the smoothness bound t is presented. Three instances of the HCDLP were solved for genus 31 curves over $\mathbb{F}_{2^2}$, $\mathbb{F}_{2^3}$, and $\mathbb{F}_{2^4}$, the last being out of the range of applicability of Pollard's rho method. In addition, convincing evidence that the HCDLP for a genus 31 curve over $\mathbb{F}_{2^5}$ is tractable (in particular, requires less time than cracking DES) is presented.

6 Open problems

Further experiments with implementing genus 2 and 3 hyperelliptic curve arithmetic on various platforms are needed in order to judge the true merits of hyperelliptic curve cryptosystems over competing schemes such as elliptic curve systems and RSA.

The potential of sieving for relation-generation in index-calculus HCDLP algorithms has yet to be fully explored. In particular, there is much room for improvement to the methods in [21]. For example, by sieving over polynomials corresponding to divisors which are the sum of only a few prime divisors from the factor base, sparse relations (few non-zero entries) will be generated. The algorithm in [21] suffers from the fact that the relations generated are dense, resulting in a dense relation matrix and much slower linear algebra. In addition, many of the techniques from factoring such as self-initialization and the large prime variant can be applied easily in the HCDLP context. These topics are currently under investigation.

Can the real quadratic representation be used to speed up cryptographic schemes? For instance, subexponential methods can profit from the baby step operation since it generates a new reduced ideal, that behaves like a random reduced ideal, with much less effort than Cantor's algorithm. Also, it is still an open problem to improve the arithmetic on real quadratic Koblitz curves.

Another important open question is whether asymptotically faster index-calculus algorithms exist for solving the HCDLP. In particular, is there an algorithm with runtime $O(L_{q^g}[1/3, c])$, where

$$L_n[1/3, c] = \exp\left((c + o(1))(\log N)^{1/3}(\log\log N)^{2/3}\right) \; ?$$

The number field sieve algorithms for integer factorization and discrete logarithm computation in finite fields have similar runtimes, but it is unknown whether ideas from these algorithms can be applied to improve on the asymptotic runtimes of HCDLP index-calculus algorithms. Although the ideas from some factoring algorithms have been successfully applied to the discrete logarithm problem in quadratic number fields (which is closely related to the HCDLP), it is even unknown if ideas from the number field sieve can be applied in that case. The existence of such an algorithm for the HCDLP would be especially noteworthy, because it most likely would imply that the ECDLP can be solved in subexponential time [9].

Galbraith [25] has shown that Weil descent can be used to attack the HCDLP on some low genus hyperelliptic curves defined over characteristic two finite fields of composite extension degrees. The classification and enumeration of those hyperelliptic curves susceptible to the Weil descent attack is an open problem.

Maximum resistance to Pollard's rho algorithm is provided by hyperelliptic curves C for which $\#J_C(\mathbb{F}_q)$ is divisible by a large prime. Thus efficient algorithms for computing $\#J_C(\mathbb{F}_q)$ are crucial when selecting curves for cryptographic applications. The problem of determining $\#J_C(\mathbb{F}_q)$ for elliptic curves C is now considered to be well-solved—see Schoof's algorithm [68] and its derivatives when the characteristic of $\mathbb{F}_q$ is odd, and Satoh's algorithm [65] (as modified in [22] and [72]) when the characteristic of $\mathbb{F}_q$ is even. Kedlaya's algorithm [42] (see also [82]) determines $\#J_C(\mathbb{F}_q)$ for hyperelliptic curves C of genus $g \geq 2$ over finite fields of small characteristic. The AGM method of Mestre and Harley (see [32]) is very efficient for genus 1 and 2 curves over characteristic two finite fields. However, the extensions of Schoof's algorithm for determining $\#J_C(\mathbb{F}_q)$ when C has genus $g \geq 2$ and $\mathbb{F}_q$ has large characteristic [61, 4] are currently very slow for parameters of sizes that are of cryptographic interest (see [33] for the most recent work and overview). Determining efficient (and practical) algorithms for the case where $\mathbb{F}_q$ is a prime field and $g = 2, 3$ or 4, is an outstanding problem in this area. We note that the CM method (see e.g. [85]) can be used to efficiently select cryptographically suitable genus 2 curves over fields of any characteristic.

Acknowledgements

We would like to thank the anonymous referees for their numerous constructive comments.

References

[1] C. Abel, *Ein Algorithmus zur Berechnung der Klassenzahl und des Regulators reell-quadratischer Ordnungen,* Ph.D. Thesis, Universität des Saarlandes, Saarbrücken, 1994.

[2] L. Adleman and J. DeMarrais, "A subexponential algorithm for discrete logarithms over all finite fields", *Mathematics of Computation*, **61** (1993), 1-15.

[3] L. Adleman, J. DeMarrais and M. Huang, "A subexponential algorithm for discrete logarithms over the rational subgroup of the jacobians of large genus hyperelliptic curves over finite fields", *Algorithmic Number Theory*, LNCS **877**, 1994, 28-40.

[4] L. Adleman and M. Huang, "Counting rational points on curves and abelian varieties over finite fields", *Algorithmic Number Theory—ANTS-II*, LNCS **1122**, 1996, 1-16.

[5] S. Arita, "Construction of secure C_{ab} curves using modular curves", *Algorithmic Number Theory—ANTS-IV*, LNCS **1838**, 2000, 113-126.

[6] E. Artin, "Quadratische körper im Gebiete der höheren kongruenzen I, II", *Math. Zeitschr.*, **19** (1924), 153-206.

[7] E. Barreiro, J. Sarlabous and J. Cherdieu, "Efficient reduction on the jacobian variety of Picard curves", *Coding Theory, Cryptography and Related Areas*, Springer-Verlag, 2000, 13-28.

[8] M. Bauer, "A subexponential algorithm for solving the discrete logarithm problem in the Jacobian of high genus hyperelliptic curves over arbitrary finite fields, preprint, 1999.

[9] M. Bauer and S. Hamdy, "On class group computations using the number field sieve", preprint, 2003.

[10] N. Boston, T. Clancy, Y. Liow and J. Webster, "Genus two hyperelliptic curve coprocessor", *Cryptographic Hardware and Embedded Systems—CHES 2002*, LNCS **2523**, 2002, 400-414.

[11] J. Buchmann, "A subexponential algorithm for the determination of class groups and regulators of algebraic number fields", *Séminaire de Théorie des Nombres, Paris 1988-1989*, Progress in Mathematics, Birkhäuser, 1990, 27-41.

[12] D. Cantor, "Computing in the jacobian of a hyperelliptic curve", *Mathematics of Computation*, **48** (1987), 95-101.

[13] D. Coppersmith, "Fast evaluation of logarithms in fields of characteristic two", *IEEE Transactions on Information Theory*, **30** (1984), 587-594.

[14] W. Diffie and M. Hellman, "New directions in cryptography", *IEEE Transactions on Information Theory*, **22** (1976), 644-654.

[15] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms", *IEEE Transactions on Information Theory*, **31** (1985), 469-472.

[16] A. Enge, *Hyperelliptic Cryptosystems*, Ph.D. thesis, Universität Augsburg, 2000.

[17] A. Enge, "Computing discrete logarithms in high-genus hyperelliptic jacobians in provably subexponential time", *Mathematics of Computation*, **71** (2001), 729-742.

[18] A. Enge, "A general framework for subexponential discrete logarithm algorithms in groups of unknown order", *Finite Geometries, Developments in Mathematics vol. 3*, Kluwer Academic Publishers, Dordrecht 2001, 133-146.

[19] A. Enge and P. Gaudry, "A general framework for subexponential discrete logarithm algorithms", *Acta Arithmetica*, **102** (2002), 83-103.

[20] A. Enge and A. Stein, "Smooth ideals in hyperelliptic function fields", *Mathematics of Computation*, **71** (2001), 1219-1230.

[21] R. Flassenberg and S. Paulus, "Sieving in function fields", *Experimental Mathematics*, **8** (1999), 339-349.

[22] M. Fouquet, P. Gaudry and R. Harley, "An extension of Satoh's algorithm and its implementation", *Journal of the Ramanujan Mathematical Society*, **15** (2000), 281-318.

[23] G. Frey, "Applications of arithmetical geometry to cryptographic constructions", *Proceedings of the Fifth International Conference on Finite Fields and Applications*, Springer-Verlag, 2001, 128-161.

[24] G. Frey and H. Rück, "A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves", *Mathematics of Computation*, **62** (1994), 865-874.

[25] S. Galbraith, "Weil descent of Jacobians", *Discrete Applied Mathematics*, to appear.

[26] S. Galbraith, F. Hess and N. Smart, "Extending the GHS Weil descent attack", *Advances in Cryptology—EUROCRYPT 2002*, LNCS **2332**, 2002, 29-44.

[27] S. Galbraith, S. Paulus and N. Smart, "Arithmetic on superelliptic curves", *Mathematics of Computation*, **71** (2002), 393-405.

[28] R. Gallant, R. Lambert and S. Vanstone, "Improving the parallelized Pollard lambda search on anomalous binary curves", *Mathematics of Computation*, **69** (2000), 1699-1705.

[29] R. Gallant, R. Lambert and S. Vanstone, "Faster point multiplication on elliptic curves with efficient endomorphisms", *Advances in Cryptology—CRYPTO 2001*, LNCS **2139** (2001), 190-200.

[30] J. von zur Gathen and K. Ma, "Analysis of Euclidean algorithms for polynomials over finite fields", *Journal of Symbolic Computation*, **9** (1990), 429-455.

[31] P. Gaudry, "An algorithm for solving the discrete log problem on hyperelliptic curves", *Advances in Cryptology—EUROCRYPT 2000*, LNCS **1807**, 2000, 19-34.

[32] P. Gaudry, "A comparison and a combination of SST and AGM algorithms for counting points of elliptic curve in characteristic 2", *Advances in Cryptology—ASIACRYPT 2002*, LNCS **2501** (2002), 311-327.

[33] P. Gaudry and R. Harley, "Counting points on hyperelliptic curves over finite fields", *Algorithmic Number Theory—ANTS-IV*, LNCS **1838**, 2000, 313-332.

[34] P. Gaudry, F. Hess and N. Smart, "Constructive and destructive facets of Weil descent on elliptic curves", *Journal of Cryptology*, **15** (2002), 19-46.

[35] D. Gordon, "Discrete logarithms in $GF(p)$ using the number field sieve", *SIAM Journal on Discrete Mathematics*, **6** (1993), 124-138.

[36] C. Günther, T. Lange and A. Stein, "Speeding up the arithmetic on Koblitz curves of genus two", *Selected Areas in Cryptography—SAC 2000*, LNCS **2012**, 2001, 106-117.

[37] J. Hafner and K. McCurley, "A rigorous subexponential algorithm for computation of class groups", *Journal of the American Mathematical Society*, **2** (1989), 837-850.

[38] F. Hess, "The GHS attack revisited", *Advances in Cryptology—EUROCRYPT 2003*, LNCS **2656** (2003), 374-387.

[39] M. Huang and D. Ierardi, "Efficient algorithms for the Riemann-Roch problem and for addition in the jacobian of a curve", *Journal of Symbolic Computation*, **18** (1994), 519-539.

[40] M. Jacobson, A. Menezes and A. Stein, "Solving elliptic curve discrete logarithm problems using Weil descent", *Journal of the Ramanujan Mathematical Society*, **16** (2001), 231-260.

[41] M. Jacobson and A. van der Poorten, "Computational aspects of NUCOMP", *Algorithmic Number Theory—ANTS-IV*, LNCS **2369**, 2002, 120-133.

[42] K. Kedlaya, "Counting points on hyperelliptic curves using Monsky-Washnitzer cohomology", *Journal of the Ramanujan Mathematical Society*, **16** (2001), 323-338.

[43] N. Koblitz, "Elliptic curve cryptosystems", *Mathematics of Computation*, **48** (1987), 203-209.

[44] N. Koblitz, "Hyperelliptic cryptosystems", *Journal of Cryptology*, **1** (1989), 139-150.

[45] N. Koblitz, "CM-curves with good cryptographic properties", *Advances in Cryptology—CRYPTO '91*, LNCS **576**, 1992, 279-287.

[46] T. Lange, *Fast Arithmetic on Hyperelliptic Curves*, Ph.D. thesis, Universität Gesamthochschule Essen, 2002.

[47] T. Lange, "Efficient arithmetic on genus 2 hyperelliptic curves over finite fields via explicit formulae", *Cryptology ePrint Archive: Report 2002/121*, 2002.

[48] M. Maurer, A. Menezes and E. Teske, "Analysis of the GHS Weil descent attack on the ECDLP over characteristic two finite fields of composite degree", *LMS Journal of Computation and Mathematics*, **5** (2002), 127-174.

[49] A. Menezes and M. Qu, "Analysis of the Weil descent attack of Gaudry, Hess and Smart", *Topics in Cryptology—CT-RSA 2001*, LNCS **2020**, 2001, 308-318.

[50] A. Menezes, E. Teske and A. Weng, "Weak fields for ECC", preprint, 2003.

[51] A. Menezes, Y. Wu and R. Zuccherato, "An elementary introduction to hyperelliptic curves", appendix in *Algebraic Aspects of Cryptography* by N. Koblitz, Springer-Verlag, 1998, 155-178.

[52] V. Miller, "Uses of elliptic curves in cryptography", *Advances in Cryptology—CRYPTO '85*, LNCS **218**, 1986, 417-426.

[53] V. Müller, A. Stein and C. Thiel, "Computing discrete logarithms in real quadratic congruence function fields of large genus", *Mathematics of Computation*, **68** (1999), 807-822.

[54] V. Nechaev, "Complexity of a determinate algorithm for the discrete logarithm problem", *Mathematical Notes*, **55** (1994), 165-172.

[55] P. van Oorschot and M. Wiener, "Parallel collision search with cryptanalytic applications", *Journal of Cryptology*, **12** (1999), 1-28.

[56] Y. Park, S. Jeong and J. Lim, "Speeding up point multiplication on hyperelliptic curves with efficiently-computable endomorphisms", *Advances in Cryptology—EUROCRYPT 2002*, LNCS **2332** (2002), 197-208.

[57] S. Paulus, "An algorithm of subexponential type computing the class group of quadratic orders over principal ideal domains", *Algorithmic Number Theory—ANTS-II*, LNCS **1122**, 1996, 247-262.

[58] S. Paulus and H.-G. Rück, "Real and imaginary quadratic representations of hyperelliptic function fields", *Mathematics of Computation*, **68** (1999), 1233-1241.

[59] S. Paulus and A. Stein, "Comparing real and imaginary arithmetics for divisor class groups of hyperelliptic curves", *Algorithmic Number Theory—ANTS-III*, LNCS **1423**, 1998, 576-591.

[60] J. Pelzl, T. Wollinger, J. Guajardo and C. Paar, "Hyperelliptic curve cryptosystems: Closing the performance gap to elliptic curve (update)", *Cryptology ePrint Archive: Report 2003/026*, 2003.

[61] J. Pila, "Frobenius maps of abelian varieties and finding roots of unity in finite fields", *Mathematics of Computation*, **55** (1990), 745-763.

[62] S. Pohlig and M. Hellman, "An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance", *IEEE Transactions on Information Theory*, **24** (1978), 106-110.

[63] J. Pollard, "Monte Carlo methods for index computation mod p", *Mathematics of Computation*, **32** (1978), 918-924.

[64] H. Rück, "On the discrete logarithm problem in the divisor class group of curves", *Mathematics of Computation*, **68** (1999), 805-806.

[65] T. Satoh, "The canonical lift of an ordinary elliptic curve over a prime field and its point counting", *Journal of the Ramanujan Mathematical Society*, **15** (2000), 247-270.

[66] T. Satoh and K. Araki, "Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves", *Commentarii Mathematici Universitatis Sancti Pauli*, **47** (1998), 81-92.

[67] R. Scheidler, A. Stein, and H. C. Williams. "Key-exchange in real quadratic congruence function fields", *Designs, Codes and Cryptography*, **7** (1996), 153-174.

[68] R. Schoof, "Elliptic curves over finite fields and the computation of square roots mod p", *Mathematics of Computation*, **44** (1985), 483-494.

[69] I. Semaev, "Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p", *Mathematics of Computation*, **67** (1998), 353-356.

[70] D. Shanks, On Gauss and composition I, II. In *Proc. NATO ASI on Number Theory and Applications*, Kluwer Academic Press 1989, 163–204.

[71] V. Shoup, "Lower bounds for discrete logarithms and related problems", *Advances in Cryptology—EUROCRYPT '97*, LNCS **1233** (1997), 256-266.

[72] B. Skjernaa, "Satoh's algorithm in characteristic 2", *Mathematics of Computation*, **72** (2003), 477-487.

[73] N. Smart, "Experiments using an analogue of the number field sieve algorithm to solve the discrete logarithm problem in the Jacobians of hyperelliptic curves", HP Laboratories Bristol, Technical Report HPL-97-130, 1997.

[74] N. Smart, "The discrete logarithm problem on elliptic curves of trace one", *Journal of Cryptology*, **12** (1999), 193-196.

[75] J. Solinas, "Efficient arithmetic on Koblitz curves", *Designs, Codes and Cryptography*, **19** (2000), 195-249.

[76] A. Stein, "Equivalences between elliptic curves and real quadratic congruence function fields", *Journal de Theorie des Nombres de Bordeaux*, **9** (1997), 75-95.

[77] A. Stein, "Sharp upper bounds for arithmetics in hyperelliptic function fields", *Journal of the Ramanujan Mathematical Society*, **9** (2001), 1-86.

[78] A. Stein and E. Teske, "Explicit bounds and heuristics on class numbers in hyperelliptic function fields", *Mathematics of Computation*, **71** (2002), 837-861.

[79] H. Stichtenoth, *Algebraic Function Fields and Codes*, Springer-Verlag, Berlin, 1993.

[80] E. Teske, "Speeding up Pollard's rho method for computing discrete logarithms", *Algorithmic Number Theory*, LNCS **1423**, 1998, 541-554.

[81] N. Thériault, "Index calculus attack for hyperelliptic curves of small genus", preprint, 2003.

[82] F. Vercauteren, "Computing zeta functions of hyperelliptic curves over finite fields of characteristic 2", *Advances in Cryptology—CRYPTO 2002*, LNCS **2442** (2002), 369-384.

[83] E. Volcheck, "Computing in the jacobian of a plane algebraic curve", *Algorithmic Number Theory*, LNCS **877**, 1994, 221-233.

[84] U. Vollmer, "Asymptotically fast discrete logarithms in quadratic number fields", *Algorithmic Number Theory—ANTS-IV*, LNCS **1838**, 2000, 581-594.

[85] A. Weng, "Constructing hyperelliptic curves of genus 2 suitable for cryptography", *Mathematics of Computation*, **72** (2003), 435-458.

[86] M. Wiener and R. Zuccherato, "Faster attacks on elliptic curve cryptosystems", *Selected Areas in Cryptography*, LNCS **1556**, 1999, 190-200.

[87] R. Zuccherato, "The equivalence between elliptic curve and quadratic function field discrete logarithms in characteristic 2", *Algorithmic Number Theory—ANTS-III*, LNCS **1423**, 1998, 621-638.