Generating Genus two Hyperelliptic Curves over Large Characteristic Finite Fields

Takakazu Satoh*

Department of Mathematics,
Tokyo Institute of Technology, Tokyo, 152-8551, Japan
satohcgn@mathpc-satoh.math.titech.ac.jp

Abstract. In hyperelliptic curve cryptography, finding a suitable hyperelliptic curve is an important fundamental problem. One of necessary conditions is that the order of its Jacobian is a product of a large prime number and a small number. In the paper, we give a probabilistic polynomial time algorithm to test whether the Jacobian of the given hyperelliptic curve of the form $Y^2 = X^5 + uX^3 + vX$ satisfies the condition and, if so, to give the largest prime factor. Our algorithm enables us to generate random curves of the form until the order of its Jacobian is almost prime in the above sense. A key idea is to obtain candidates of its zeta function over the base field from its zeta function over the extension field where the Jacobian splits.

Key words: hyperelliptic curve, point counting

1 Introduction

In (hyper)elliptic curve cryptography, point counting algorithms are very important to exclude the weak curves. For elliptic curves over finite fields, the SEA algorithm (see Schoof[36] and Elkies[8]) runs in polynomial time (with respect to input size). In case that the characteristic of the coefficient field is small, there are even faster algorithms based on the p-adic method (see e.g. Vercauteren[40] for a comprehensive survey). The p-adic method gives quite efficient point counting algorithms for higher dimensional objects, e.g. Kedlaya[22], Lercier and Lubicz[26], Lauder[23]. However, so far, there is no known efficient practical algorithm for hyperelliptic curves of genus two in case that the characteristic of the coefficient field is large.

In theory, Pila[32] generalized the Schoof algorithm to a point counting algorithm for Abelian varieties over finite fields. Nevertheless, a pure polynomial time algorithm has not been successfully used even for the Jacobian of hyperelliptic curves of genus two. Current implementations for crypto size curves of genus two more or less contain the BSGS process, hence the running time grows

* The work was supported by the Grant-in-Aid for the Scientific Research (B) 18340005.

exponentially. For details on implementation, see Matsuo, Chao and Tsujii[27], Gaudry and Schost[17, 18] and references cited there.

On the other hand, using basic properties on character sums (see e.g. Berndt, Evans and Williams[3] for standard facts on the topic), Furukawa, Kawazoe and Takahashi[11] gives an explicit formula for the order of Jacobians of curves of type $Y^2 = X^5 + aX$ where $a \in \mathbf{F}_p^\times$. However, there are, at most, only 8 isomorphism classes over $\mathbf{F}_p$ among these curves for each prime p. In order to obtain a curve suitable for cryptography, they try various values of p until a suitable curve is found. Their method relies on the binomial expansion of $X^5 + aX$. The idea was generalized to hyperelliptic curves over prime fields of the form $Y^2 = X^5 + a$ (ibid.) and $Y^2 = X^{2k+1} + aX$ in Haneda, Kawazoe and Takahashi[19]. Recently, Anuradha[2] obtained similar formulae for non-prime fields. But their method seems to be applicable only to binomials in X.

In this paper, we consider an intermediate case: our curve is in a certain special form but not as special as was considered in the above papers and time complexity to test one curve is of probabilistic polynomial time. More specifically, we give an algorithm to test whether the order of the Jacobian of a given hyperelliptic curve of genus two in the form $Y^2 = X^5 + uX^3 + vX$ has a large prime factor. If so, the algorithm outputs the prime factor. Moreover, under a certain condition (see Remark 5) which is always satisfied in cryptographic applications, our algorithm determines the group order itself. Numerical experiments suggest that curves in this form which are suitable for cryptography exist for any large p. We may attempt to choose the base field for which arithmetic operations can be efficiently performed.

The order of the Jacobian of hyperelliptic curves $Y^2 = X(X^{2n} + uX^n + v)$ over a prime field are already studied by Leprévost and Morain[24]. In case of $n = 2$, they gave some explicit formulae for the order of the Jacobian in terms of certain modular functions, whose evaluations are computationally feasible only for special combinations of u and v.

Our method is totally different from the preceding point counting works. Let $p \geq 5$ be a prime and let q be a power of p. However, it is recommended to take $q = p$ to avoid a possible attack due to Gaudry[15] (cf. Section 8). Let $C/\mathbf{F}_q : Y^2 = X^5 + uX^3 + vX$ be an arbitrary (in view of cryptographic application, randomly) given hyperelliptic curve. We now observe a key idea of our method. Put $r = q^4$. We denote the Jacobian variety of C by J, which is an Abelian variety of dimension two defined over $\mathbf{F}_q$. Now J is $\mathbf{F}_r$-isogenous to a square of an elliptic curve defined over $\mathbf{F}_r$. Hence the (one dimensional part of the) zeta function of J as an Abelian variety over $\mathbf{F}_r$ is a square of the zeta function of the elliptic curve, which is computed by the SEA algorithm. On the other hand, we have some relation between the zeta function of J over $\mathbf{F}_r$ and over $\mathbf{F}_q$. This gives (at most 26) possible orders of $J(\mathbf{F}_q)$. For each candidate, we first check that the order is not weak for cryptographic use, that is, the order is a product of a small positive integer and a large prime. If the curve is weak, we stop here. (Note that, if the curve passes the test, its Jacobian is simple over $\mathbf{F}_q$.) Then, we take random points of $J(\mathbf{F}_q)$ to see whether the prime actually divides

$\#J(\mathbf{F}_q)$. Our algorithm runs in probabilistic polynomial time in $\log q$. In order to find a hyperelliptic curve suitable for cryptography, we repeat the process with randomly given u and v until we obtain a curve with desired properties.

In the case of characteristics two, Hess, Seroussi and Smart[20] proposed an algorithm using Weil descent to construct verifiably randomly a hyperelliptic curve in a certain family which is suitable for a hyperelliptic cryptosystem. The efficiency of their algorithm in case of large characteristics is not clear. As to products of elliptic curves, Scholten[35] randomly constructs a hyperelliptic curve of genus two for odd prime fields whose Jacobian is isogenous to the Weil restriction of elliptic curves. Both of the works start with elliptic curves while our algorithm starts with the hyperelliptic curve $Y^2 = X^5 + uX^3 + vX$ where u and v are given.

Recently Sutherland[38] proposed an algorithm based on a generic group model to produce Abelian varieties from random hyperelliptic curves. When applied to curves of genus two, its running time is quite practical but its heuristic time complexity is of sub-exponential. Although these two algorithms [20, 38] take random input data, it is highly non-trivial to observe distribution of output of the algorithm. In case of our algorithm, it is obvious that our algorithm generates each curve of type $Y^2 = X^5 + uX^3 + vX$, suitable to cryptography with equal probability if we generate random u and v uniformly.

The Jacobian has complex multiplications by $\sqrt{-1}$ for all u and v as long as the curve is actually a hyperelliptic curve. This fact can be used to make scalar multiplication faster by the Gallant, Lambert and Vanstone algorithm[12]. We can also take advantage of real multiplications with efficient evaluations (if any) due to Takashima[39]. However we also note that such an efficient endomorphism also speeds up solving discrete log problems Duursma, Gaudry and Morain[7].

In case of $q \equiv 1 \bmod 4$, we can apply our algorithm to $Y^2 = X(X^2 - \alpha^2)(X^2 - \beta^2)$ for randomly generated $\alpha, \beta \in \mathbf{F}_q^\times$. (When $q \equiv 3 \bmod 4$, such curves are never suitable for cryptography. See Remark 2.) All 2-torsion points of the Jacobian of such a curve are $\mathbf{F}_q$-rational. Hence, the efficient scalar multiplication algorithm due to Gaudry[14] is applicable to them. Although it is not clear how often such a curve is suitable for cryptography, a numerical experiment found a few suitable curves.

After the initial submission for the conference, the author learned that Gaudry and Schost[16] completely describes the hyperelliptic curves whose Jacobian is isogenous to a product of two elliptic curves by an isogeny with kernel $\mathbf{Z}/2\mathbf{Z} \oplus \mathbf{Z}/2\mathbf{Z}$. Our idea is probably applicable to such curves given by in terms of the Rosenhain form or the invariants (Ω, Υ) in the notation of [16]. The j-invariants of two elliptic curves will be different but no essential change to our idea is required. However, the author have not derived explicit formulae for these parameters, yet. Recently, Paulhus[31] gave explicit descriptions of splitting of Jacobian into elliptic curves for some curves of genus greater than two. Although the use of curves with genues greater than four in cryptography is rather questionable in view of the index calculus method due to Gaudry[13], it would be interesting to consider number theoretic properties of the order of such curves.

The rest of the paper is organized as follows. In Section 2, we review some facts on arithmetic properties of the Jacobian varieties. In Section 3, we give an explicit formula of the elliptic curve whose power is isogenous to the Jacobian of the given hyperelliptic curve of the above form. In Section 4, we show how to retrieve possible orders of the Jacobian via the decomposition obtained in the preceding section. In Section 5, we state our algorithm and observe its computational complexity. In Section 6, we give an illustrative small numerical example. In Section 7, we report results of numerical experiments with cryptographic size parameters. We observe some security implication of our curves in section 8.

Throughout the paper, we let p be a prime greater than or equal to 5 and q a power of p. We put $r = q^4$.

2 Some properties of the Jacobian varieties

We summarize arithmetic properties of the Jacobian varieties used in the later sections. See the surveys Milne[29, 30] for more descriptions.

Let $A/\mathbf{F}_q$ be an Abelian variety of dimension d. The (one dimensional part of the) zeta function $Z_A(T, \mathbf{F}_q)$ is the characteristic polynomial of the q-th power Frobenius map on $V_l(A) = T_l(A) \otimes_{\mathbf{Z}_l} \mathbf{Q}_l$ where l is a prime different from p and $T_l(A)$ is the l-adic Tate module. It is known that $Z_A(T, \mathbf{F}_q) \in \mathbf{Z}[T]$ with $\deg Z_A(T, \mathbf{F}_q) = 2d$ and that it is independent of the choice of l. It holds that

$$\#A(\mathbf{F}_q) = Z_A(1, \mathbf{F}_q). \tag{1}$$

Let $\prod_{i=1}^{2d}(T - z_{i,q})$ be the factorization of $Z_A(T, \mathbf{F}_q)$ in $\mathbf{C}[T]$. Permuting indices if necessary, we may assume that

$$z_{1,q}z_{2,q} = q, \ \ \ldots, \ \ z_{2d-1,q}z_{2d,q} = q. \tag{2}$$

Let $n \in \mathbf{N}$ and put $\tilde{q} = q^n$. Since the $\tilde{q}$-th power map is the n-times iteration of the q-th power map, we see

$$\{z_{1,\tilde{q}}, z_{2,\tilde{q}}, \ldots, z_{2d,\tilde{q}}\} = \{z_{1,q}^n, z_{2,q}^n, \ldots, z_{2d,q}^n\} \tag{3}$$

including multiplicity. It holds that $|z_{i,\tilde{q}}| = \sqrt{\tilde{q}}$.

In case that A is isogenous to $A_1 \times A_2$ over $\mathbf{F}_q$, we have $V_l(A) \cong V_l(A_1) \oplus V_l(A_2)$ as $\mathrm{Gal}(\overline{\mathbf{F}_q}/\mathbf{F}_q)$-modules. Hence

$$Z_A(T, \mathbf{F}_q) = Z_{A_1}(T, \mathbf{F}_q)Z_{A_2}(T, \mathbf{F}_q). \tag{4}$$

Let $E/\mathbf{F}_q$ be an elliptic curve, which is an Abelian variety of dimension 1 over $\mathbf{F}_q$. The above items translate to the well known formula

$$Z_E(T, \mathbf{F}_q) = T^2 - tT + q$$

with $|t| \leq 2\sqrt{q}$ where $t = q + 1 - \#E(\mathbf{F}_q)$.

Let $C/\mathbf{F}_q$ be a hyperelliptic curve of genus two and let J be its Jacobian variety, which is an Abelian variety of dimension two defined over $\mathbf{F}_q$. Let $z_{1,q}$, ..., $z_{4,q}$ be the roots of $Z_J(T, \mathbf{F}_q)$ arranged as in (2). We see

$$Z_J(T, \mathbf{F}_q) = T^4 - a_q T^3 + b_q T^2 - q a_q T + q^2 \tag{5}$$

where

$$a_q = \sum_{i=1}^{4} z_{i,q}, \qquad b_q = \sum_{i=1}^{3} \sum_{j=i+1}^{4} z_{i,q} z_{j,q}.$$

We note $|a_q| \leq 4\sqrt{q}$ and $|b_q| \leq 6q$. We will also use the Hasse-Weil bound

$$(\sqrt{q} - 1)^4 \leq \#J_C(\mathbf{F}_q) \leq (1 + \sqrt{q})^4. \tag{6}$$

3 Decomposition of the Jacobian

In this section, we give an explicit formula of two elliptic curves whose product is isogenous to the Jacobian of the hyperelliptic curves of our object. Such a decomposition has been more or less known, e.g. Leprévost and Morain[24], Cassel and Flynn[5, Chap. 14], and Frey and Kani[9]. Here we derive a formula which is ready to implement our method efficiently.

Let $C : Y^2 = X^5 + uX^3 + vX$ be a hyperelliptic curve where $u \in \mathbf{F}_q$ and $v \in \mathbf{F}_q^\times$. We denote the Jacobian variety of C by J. There exist α, $\beta \in \mathbf{F}_r^\times$ such that

$$X^5 + uX^3 + vX = X(X^2 - \alpha^2)(X^2 - \beta^2).$$

We choose and fix $s \in \mathbf{F}_{q^8}^\times$ satisfying $s^2 = \alpha\beta$. In fact, $s \in \mathbf{F}_r^\times$ since $s^4 = \alpha^2\beta^2 = v \in \mathbf{F}_q^\times$. It is straightforward to verify

$$\begin{aligned} X^2 + (\alpha + \beta)X + \alpha\beta &= A(X + s)^2 + B(X - s)^2 \\ X^2 - (\alpha + \beta)X + \alpha\beta &= B(X + s)^2 + A(X - s)^2 \end{aligned}$$

where

$$\begin{aligned} A &= \frac{1}{2}\left(1 + \frac{\alpha + \beta}{2s}\right), \\ B &= \frac{1}{2}\left(1 - \frac{\alpha + \beta}{2s}\right). \end{aligned}$$

Then

$$\begin{aligned} X^4 + uX^2 + v &= (X^2 - \alpha^2)(X^2 - \beta^2) = (X + \alpha)(X + \beta)(X - \alpha)(X - \beta) \\ &= AB\left((X + s)^4 + \left(\frac{B}{A} + \frac{A}{B}\right)(X + s)^2(X - s)^2 + (X - s)^4\right). \end{aligned}$$

Note that X can be rewritten in terms of $X - s$ and $X + s$:

$$X = \frac{1}{4s}\left((X + s)^2 - (X - s)^2\right).$$

Define $E_1/\mathbf{F}_r$ and $E_2/\mathbf{F}_r$ by

$$\begin{aligned} E_1 \ : \ Y^2 &= \ \delta(X-1)(X^2-\gamma X+1) \\ E_2 \ : \ Y^2 &= \ -\delta(X-1)(X^2-\gamma X+1) \end{aligned}$$

where

$$\delta = \ \frac{AB}{4s} = -\frac{(\alpha-\beta)^2}{64s^3}, \tag{7}$$

$$\gamma = \ -\left(\frac{B}{A}+\frac{A}{B}\right) = 2(\alpha^2+6\alpha\beta+\beta^2)/(\alpha-\beta)^2. \tag{8}$$

Then, we have two covering maps $\varphi_i : C \to E_i$ defined over $\mathbf{F}_r$ by

$$\begin{aligned} \varphi_1(x,y) &= \ \left(\left(\frac{x+s}{x-s}\right)^2, \frac{y}{(x-s)^3}\right), \\ \varphi_2(x,y) &= \ \left(\left(\frac{x-s}{x+s}\right)^2, \frac{y}{(x+s)^3}\right). \end{aligned}$$

They induce maps $\varphi_i^* : \mathrm{Div}(E_i) \to \mathrm{Div}(C)$ and $\varphi_{i*} : \mathrm{Div}(C) \to \mathrm{Div}(E_i)$. They again induce maps (which are also denoted by) $\varphi_i^* : \mathrm{Pic}^0(E_i) \ (\cong E) \to \mathrm{Pic}^0(C) \ (\cong J)$ and $\varphi_{i*} : J \to E_i$. We note that $\varphi_{i*} \circ \varphi_i^*$ is the multiplication by 2 map on E_i and that both $\varphi_{1*} \circ \varphi_2^*$ and $\varphi_{2*} \circ \varphi_1^*$ are the zero maps. Therefore J is isogenous to $E_1 \times E_2$.

Since $2|[\mathbf{F}_r : \mathbf{F}_p]$ and $p \geq 5$, both E_1 and E_2 are isomorphic to the following elliptic curve in the short Weierstrass from:

$$E \ : \ Y^2 = X^3 - \frac{(\gamma-2)(\gamma+1)}{3}\delta^2 X - \frac{(\gamma-2)^2(2\gamma+5)}{27}\delta^3. \tag{9}$$

Eventually, J is isogenous to $E \times E$ over $\mathbf{F}_r$. Using (4), we conclude

$$Z_J(T, \mathbf{F}_r) = Z_E(T, \mathbf{F}_r)^2. \tag{10}$$

Remark 1. Observe that in fact $\gamma \in \mathbf{F}_{q^2}$. By (8), we see that $\gamma \in \mathbf{F}_q$ if and only if either $u = 0$ or $u \neq 0$ and $\alpha\beta \in \mathbf{F}_q^\times$ (i.e. v is a square element in $\mathbf{F}_q$). Thus, we do not need to run the SEA algorithm to $E/\mathbf{F}_r$. Define $E'/\mathbf{F}_q(\gamma)$ by

$$E' \ : \ Y^2 = X^3 - \frac{(\gamma-2)(\gamma+1)}{3}X - \frac{(\gamma-2)^2(2\gamma+5)}{27}. \tag{11}$$

Let σ be the trace of the $\#\mathbf{F}_q(\gamma)$-th power Frobenius endomorphism on E'. We obtain σ by running the SEA algorithm to $E'/\mathbf{F}_q(\gamma)$. Note that the size of the coefficient field is smaller than the size of $\mathbf{F}_r$ by a factor of $1/2$ or $1/4$. Let τ' be the trace of the r-th power Frobenius endomorphism on E'. Then

$$\tau' = \begin{cases} (\sigma^2-2q)^2 - 2q^2 & (\gamma \in \mathbf{F}_q), \\ \sigma^2 - 2q^2 & (\gamma \notin \mathbf{F}_q). \end{cases}$$

Now E is isomorphic to E' over $\mathbf{F}_{r^2}$. Let τ be the trace of the r-th power Frobenius endomorphism on E. Unless $j(E) = 0$ or $j(E) = 1728$, we have

$$\tau = \begin{cases} \tau' & (\delta^{(r-1)/2} = 1), \\ -\tau' & (\delta^{(r-1)/2} = -1). \end{cases}$$

(And it is easy to compute τ in case of $j(E) = 0$ or $j(E) = 1728$, see e.g. Schoof[37, Sect. 4].) This device does not affect growth rate of the computational complexity of our algorithm. However practical performance improvement by this is significant, since the most of computational time is spent for the SEA algorithm.

Remark 2. Note that, in fact, E_1 and E_2 are defined over $\mathbf{F}_q(s)$. Changing the sign of α if necessary in case of $q \equiv 3 \bmod 4$, we see $s \in \mathbf{F}_q$ when v is a fourth power element of $\mathbf{F}_q^\times$. In this case J already splits over $\mathbf{F}_q$. Note that in case of $q \equiv 3 \bmod 4$, any square element in $\mathbf{F}_q^\times$ is a fourth power element.

4 Computing possible order of the Jacobian

In this section, we consider how to obtain $Z_J(T, \mathbf{F}_q)$ from $Z_J(T, \mathbf{F}_r)$. Actually, this is quite elementary.

Let $z_{1,q}, \ldots, z_{4,q}$ be the roots of $Z_J(T, \mathbf{F}_q)$ arranged as (2). Put $s_n = \sum_{i=1}^4 z_{i,q}^n$ with a convention $s_0 = 4$. Then

$$\begin{aligned} s_1 &= a_q, \\ s_2 &= a_q^2 - 2b_q, \\ s_3 &= a_q^3 - 3a_q b_q + 3qa_q \end{aligned}$$

and

$$s_i = a_q s_{i-1} - b_q s_{i-2} + q a_q s_{i-3} - q^2 s_{i-4}$$

for $i \geq 4$. In particular, we obtain

$$\begin{aligned} s_4 &= a_q^4 - 4(b_q - q)a_q^2 + 2b_q^2 - 4q^2, \\ s_8 &= a_q^8 - 8(b_q - q)a_q^6 + (20b_q^2 - 32qb_q + 4q^2)a_q^4 \\ &\quad + (-16b_q^3 + 24qb_q^2 + 16q^2 b_q - 16q^3)a_q^2 + 2b_q^4 - 8q^2 b_q^2 + 4q^4. \end{aligned}$$

Recall that $r = q^4$. Hence $a_r = s_4$ and

$$b_r = (s_4^2 - s_8)/2 = 2q^2 a_q^4 + (-4qb_q^2 + 8q^2 b_q - 8q^3)a_q^2 + b_q^4 - 4q^2 b_q^2 + 6q^4.$$

Recall that J is isogenous to $E \times E$ over $\mathbf{F}_r$ where E is defined by (11). Let t be the trace of r-th Frobenius map on E. Then, $Z_E(T, \mathbf{F}_r) = T^2 - tT + r$. Thus (10) gives

$$T^4 - a_r T^3 + b_r T^2 - \cdots = T^4 - 2tT^3 + (2r + t^2)T^2 + \cdots,$$

that is

$$\begin{aligned} 0 = \ & a_q^4 - 4(b_q - q)a_q^2 + 2b_q^2 - 4q^2 - 2t, \qquad (12) \\ 0 = \ & 2q^2 a_q^4 + (-4qb_q^2 + 8q^2 b_q - 8q^3)a_q^2 + b_q^4 - 4q^2 b_q^2 + 6q^4 - (2q^4 + t^2). \end{aligned}$$

Eliminating b_q by computing a resultant of the above two polynomials, we obtain

$$\begin{aligned} a_q^{16} - \ & 32qa_q^{14} + (368q^2 - 8t)a_q^{12} + (-1920q^3 + 64tq)a_q^{10} \\ & + (4672q^4 + 64tq^2 - 112t^2)a_q^8 + (-5120q^5 - 1024tq^3 + 768t^2 q)a_q^6 \\ & + (2048q^6 + 1024tq^4 - 512t^2 q^2 - 256t^3)a_q^4 = 0. \qquad (13) \end{aligned}$$

This yields at most 13 possible values for a_q. Note that a_q is an integer satisfying $|a_q| \leq 4\sqrt{q}$. In order to find integer solutions of a_q, we choose any prime l satisfying $l > 8\sqrt{q}$ and factor the above polynomial in $\mathbf{F}_l$. We only need linear factors. For each $\mathbf{F}_l$-root, we check whether it is a genuine root in characteristic zero and its absolute value does not exceed $4\sqrt{q}$. For each possible a_q, we easily obtain at most two possible values of b_q satisfying the above equations. Thus, we have obtained at most 26 candidates for $\#J(\mathbf{F}_q)$.

5 The algorithm and its complexity

In this section, we analyze a time computational complexity of our algorithm. First, we state our method in a pseudo-code. Then, we show our algorithm terminates in probabilistic polynomial time in $\log q$. We denote the identity element of J by 0 in the following.

In addition to the coefficients of hyperelliptic curve C, we give two more inputs: the "cofactor bound" M and a set of "test points" D, which is any subset of $J(\mathbf{F}_q)$ satisfying $\#D > M$.

In order that the discrete logarithm problem on $J(\mathbf{F}_q)$ is not vulnerable to the Pohlig-Hellman attack[33], $\#J(\mathbf{F}_q)$ must be divisible by a large prime (at least 160 bit in practice). We request that the largest prime factor is greater than $\#J(\mathbf{F}_q)/M$, which ensures that the factor is greater than $(\sqrt{q}-1)^4/M$. In the following algorithm, M must be less than $(\sqrt{q}-1)^2$. In practice, $M < 2^8$ (at most) in view of efficiency of group operation. So, building up D is easy for such small M.

Algorithm 1.
Input: Coefficients u, $v \in \mathbf{F}_q$ in $C : Y^2 = X^5 + uX^3 + vX$,
a cofactor bound $M \in \mathbf{N}$ satisfying $M < (\sqrt{q}-1)^2$,
a subset D of $J(\mathbf{F}_q)$ satisfying $\#D > M$.
Output: The largest prime factor of $\#J(\mathbf{F}_q)$ if it is greater than $\#J(\mathbf{F}_q)/M$. Otherwise, **False**.
Procedure:
1: Let α_0, $\beta_0 \in \mathbf{F}_{q^2}$ be the solution of $x^2 + ux + v = 0$.
2: Find $\alpha \in \mathbf{F}_r$ and $\beta \in \mathbf{F}_r$ satisfying $\alpha^2 = \alpha_0$, $\beta^2 = \beta_0$.

3: Compute δ and γ by (7) and (8), respectively.
4: Compute $\#E(\mathbf{F}_r)$ by the SEA algorithm and put $t = 1 + q^r - \#E(\mathbf{F}_r)$.
5: Find a prime l satisfying $8\sqrt{q} < l \leq q$.
6: Find solutions of (13) modulo l.
7: for each solution τ do:
8: Lift $\tau \in \mathbf{F}_l$ to $a_q \in \mathbf{Z}$ so that $|a_q| \leq 4\sqrt{q}$.
9: if a_q is really a root of (13) and if a_q is even then
10: for each integer solution b_q of (12) satisfying $|b_q| \leq 6q$ do:
11: $L = 1 - a_q + b_q - qa_q + q^2$ /* cf. (1), (5) */
12: if $(\sqrt{q}-1)^4 \leq L \leq (\sqrt{q}+1)^4$ then /* cf. (6) */
13: Find the largest divisor d of L less than M.
14: $L' = L/d$
15: if (L' is prime) then
16: Find a point $P \in D$ such that $dP \neq 0$.
17: if $LP = 0$, then output L' and stop.
18: endif
19: endif /* L satisfies the Hasse-Weil bound */
20: endfor /* b_q */
21: endif
22: endfor /* τ */
23: Output **False** and stop.

Remark 3. Actually, in the SEA algorithm in Step 4, we obtain t before we obtain $\#E(\mathbf{F}_q)$.

Remark 4. Instead of giving a set D by listing all points, we can specify D by some conditions and we generate elements of D during execution of the above procedure. In the implementation used in the next section, the author used

$$D = \{[P] + [Q] - 2[\infty] \ : \ P, \ Q \in C(\mathbf{F}_q) - \{\infty\}, \ P_X \neq Q_X\}$$

where ∞ is the point at infinity of C (not J) and the subscript X stands for the X-coordinate. Then, $\#D \approx O(q^2)$. It is easy to generate a uniformly random point of D in probabilistic polynomial time.

Remark 5. In case that $d \leq \frac{\sqrt{q}}{8} - \frac{1}{2}$ (which always holds when $M \leq \frac{\sqrt{q}}{8} - \frac{1}{2}$), the value of L at Step 17 gives $\#J(\mathbf{F}_q)$. The reason is as follows. Observe that

$$\left(\frac{x}{8} - \frac{1}{2}\right)((x+1)^4 - (x-1)^4) < (x-1)^4$$

for $x \in \mathbf{R}$. Thus

$$L' \geq \frac{(\sqrt{q}-1)^4}{d} \geq \frac{(\sqrt{q}-1)^4}{\frac{\sqrt{q}}{8} - \frac{1}{2}} > (\sqrt{q}+1)^4 - (\sqrt{q}-1)^4.$$

Hence, there is only one multiple of L' in the Hasse-Weil bound (6), which must be $\#J(\mathbf{F}_q)$.

Remark 6. Instead of Steps 5–6, one can choose a small prime l which does not divide the discriminant of the square free part of (13), and then factor (13) modulo l and lift the factors to modulo l^n where $l^n > 8\sqrt{q}$. See e.g. von zur Gathen and Gerhard[42] for details of both methods and comparison. In theory, both methods run in polynomial time w.r.t. input size. In practice, time for these steps are negligible for crypto size parameters. The use of a large prime is easy to implement and conceptually simple. One theoretical concern is the number of primality tests to find l. As we will see (14) below, there exist at least $\Omega(q/\log q)$ primes l satisfying $8\sqrt{q} < l < q$. Thus, average number of primality tests to find l in Step 5 is $O(\log q)$. In an actual implementation, we may search for a prime by testing odd numbers greater than $8\sqrt{q}$ one by one. Primality tests can be performed by a deterministic polynomial time algorithm by Agrawal, Kayal and Saxena[1]. Since our main interest is in hyperelliptic cryptography, we do not go into complexity arguments on primality tests further.

Theorem 1. *Let $M < (\sqrt{q}-1)^2$ be fixed. Then Algorithm 1 terminates in probabilistic polynomial time in $\log q$ (with respect to the bit operations). In case that $\#J(\mathbf{F}_q)$ is divisible by a prime greater than $(\sqrt{q}-1)^4/M$, Algorithm 1 returns the largest prime factor of $\#J(\mathbf{F}_q)$.*

Proof. Note that t, q, a_q and b_q are integers. Hence a_q must be even by (12). This explains Step 9. If we reach Step 16, we have, as an Abelian group,

$$J(\mathbf{F}_q) \cong G \oplus (\mathbf{Z}/L'\mathbf{Z})$$

where G is an Abelian group of order d. Since $\gcd(d, L') = 1$, there are at most d points $Q \in J(\mathbf{F}_q)$ satisfying $dQ = 0$. Since $\#D > M$, we can find P at Step 16 by testing at most M elements in D. Note $\#J(\mathbf{F}_q) \geq (\sqrt{q}-1)^4$. Therefore,

$$L' \geq \frac{\#J(\mathbf{F}_q)}{M} \geq \frac{\#J(\mathbf{F}_q)}{(\sqrt{q}-1)^2} \geq \sqrt{\#J(\mathbf{F}_q)}.$$

Since L' is a prime, it must be the largest prime factor of $\#J(\mathbf{F}_q)$, which completes the proof of correctness of the algorithm.

Now we consider computational complexity. Note that a bit size of any variable appearing in Algorithm 1 is bounded by $c \log q$ where c is a constant depending only on M and the primality test algorithm used in Step 5. Thus we have only to show that, instead of the number of bit operations, the number of arithmetic operations performed in Algorithm 1 is bounded by a polynomial in $\log q$. For a positive real number x, put $\theta(x) = \sum_{l \leq x} \theta(x)$ as usual where l runs over primes less than x. By Chebyshev's theorem[6], there exist constants C_1 and C_2 such that

$$Kx - C_1\sqrt{x}\log x < \theta(x) < \frac{6}{5}Kx$$

for all $x > C_2$ where $K = \log \frac{2^{1/2}3^{1/3}5^{1/5}}{30^{1/30}}$ $(= 0.921...)$. Let $\nu(q)$ be the number of primes l satisfying $8\sqrt{q} < l \leq q$. Then

$$\nu(q)\log q \geq \sum_{8\sqrt{q} < l \leq q} \log l = \theta(q) - \theta(8\sqrt{q})$$

and thus

$$\nu(q) \geq K\frac{q}{\log q} - C_3\sqrt{q} \text{ for } q > C_4 \tag{14}$$

with some C_3, $C_4 > 0$. Therefore, if we test random odd numbers between $8\sqrt{q}$ and q to find l, an average number of primality tests in Step 5 is less than $\frac{\log q}{2K}\left(1 + \frac{2C_3 \log q}{K\sqrt{q}}\right)$ for all $q > C_4$. In Steps 1, 2 and 6, we need to factor univariate polynomials over $\mathbf{F}_r$ or $\mathbf{F}_l$. However, the degree of polynomials to be factored is either 2 or 13. Hence the Cantor-Zassenhaus factorization[4] factors them in probabilistic polynomial time. The SEA algorithm (even Schoof's algorithm alone) runs in polynomial time. Summing up, we obtain our assertions. □

6 A numerical example

We give an illustrative example of our algorithm. We set $M = 16$. Let $q = p = 509$ and consider $C : Y^2 = X^5 + 3X^3 + 7X$. Then, $\mathbf{F}_r = \mathbf{F}_p(\theta)$ where $\theta^4 + 2 = 0$. For simplicity, we write an element $\mu_3\theta^3 + \mu_2\theta^2 + \mu_1\theta + \mu_0$ of $\mathbf{F}_r$ as $[\mu_3\ \mu_2\ \mu_1\ \mu_0]$. Then we have $\alpha = [193\ 0\ 90\ 0]$, $\beta = [67\ 0\ 396\ 0]$, $s = [0\ 0\ 427\ 0]$, $\delta = [29\ 0\ 488\ 0]$ and $\gamma = [0\ 56\ 0\ 17]$. Hence the short Weierstrass form of E is

$$Y^2 = X^3 + [0\ 370\ 0\ 73]X + [293\ 0\ 464\ 0].$$

An elliptic curve point counting algorithm gives $t = 126286$. We take $l = 191$. Then, (13) in this example is

$$a_q^{16} - 16288a_q^{14} + 94331520a_q^{12} - 249080786944a_q^{10} + 313906268606464a_q^8 \\ -185746793889398784a_q^6 + 41664329022490804224a_q^4 = 0.$$

Reducing modulo l, we obtain

$$\begin{aligned} 0 &= \overline{a_q}^{16} + 138\overline{a_q}^{14} + 58\overline{a_q}^{12} + 46\overline{a_q}^{10} + 63\overline{a_q}^8 + 94\overline{a_q}^6 + 136\overline{a_q}^4 \\ &= (\overline{a_q}^2 + 56\overline{a_q} + 170)(\overline{a_q}^2 + 135\overline{a_q} + 170)(\overline{a_q}^2 + 150)^2(\overline{a_q} + 28)^2(\overline{a_q} + 163)^2\overline{a_q}^4 \end{aligned}$$

where $\overline{a_q}$ is the reduction of a_q modulo l. Hence $a_q = -28$, 0, 28. In case of $a_q = -28$, Eq. (12) is $b_q^2 - 784b_q + 460992 = 0$. Thus $b_q = 1176$, 392. The former values gives $L = 274538 = 11 \cdot 24958$ and 24958 is apparently not prime. Similarly, the case $b_q = 392$ gives $L = 273754 = 13 \cdot 21058$. In case of $a_q = 0$, Eq. (12) does not have an integer solution. Finally, we consider the case $a_q = 28$. The equation (and hence the solution) for b_q is the same as that for $a_q = -28$. Now $b_q = 1176$ gives $L = 245978 = 2 \cdot 122989$ and $122989 = 29 \cdot 4241$ is not prime. But in the case of $b_q = 392$, we obtain $L = 245194 = 2 \cdot 122597$ and 122597 is prime. Take $P = (X^2 + 286X + 46,347X + 164) \in J(\mathbf{F}_p)$ written in the Mumford representation. Then, $2P = (X^2 + 365X + 23, 226X + 240) \neq 0$ but $LP = 0$. Thus, 122597 is the largest prime divisor of $\#J(\mathbf{F}_p)$. In this example, we also conclude $\#J(\mathbf{F}_p) = 245194$ because $d = 2 \leq \frac{\sqrt{509}}{8} - \frac{1}{2} = 2.32...$

7 Cryptographic size implementation

In this section, we report implementation results for cryptographic size parameters. The results show that our algorithm certainly produces hyperelliptic curves for cryptographic use with an acceptable computational complexity. In reality, almost all computation time is consumed in an elliptic curve point counting (Step 4 in Algorithm 1). We begin with remarks on an elliptic curve point counting.

First, deployment of Remark 1 is very important. Assume that we use the Karatsuba algorithm for multiplications. Then Remark 1 reduces running time by a factor of $2^{2+2\log_2 3} \approx 36.2$ in theory. More importantly, a number of modular polynomials necessary for the SEA algorithm is approximately halved.

Next the action of the Frobenius map can be computed more efficiently than a straightforward method. Let p be a prime and let n be an integer greater than 1. In this paragraph, we put $q = p^n$. Assume that we use a variant of the baby-step giant-step algorithm due to Maurer and Müller[28] to find an eigenvalue of the Frobenius endomorphism. Then a bottleneck of Elkies' point counting algorithm for elliptic curves defined over $\mathbf{F}_q$ is a computing action of the q-th power map and $\left(\frac{q-1}{2}\right)$-th power map in $\mathbf{F}_q[t]/\langle b(t)\rangle$ for some $b(t) \in \mathbf{F}_q[t]$. To reduce their computational time, we use an algorithm due to von zur Gathen and Shoup[43] with a modification to take the action of p-th power map on $\mathbf{F}_q$ into consideration. See Vercauteren[41, Sect. 3.2] in the case $p = 2$. Even in the case of $n = 2$ (which is the case we need), the modification saves much time. The overall performance improvement of elliptic curve point counting with this technique highly depends on an implementation and field parameters. In author's implementation (which uses the Elkies primes only) for $p \approx 2^{87}$ and $n = 2$, the improvement was approximately in range $20 \sim 40\%$ depending on elliptic curves.

Now we back to our algorithm. In what follows, q is the cardinality of the coefficient field of hyperelliptic curves to be generated. Assume that we need the Jacobian variety whose bit size of the group order is N. Then, $\log q \cong N/2$. Note we apply the SEA algorithm to elliptic curves over $\mathbf{F}_{q^2}$. Thus we can paraphrase that time to test one random hyperelliptic curve of the form $Y^2 = X^5+uX^3+vX$ is comparable to time to test one random elliptic curve if their group sizes are the same. Of course, this is only rough comparison. For example, in the elliptic curve case, we can employ early abort strategy (see Lercier[25, Sect. 4.1]), whereas we need to complete the SEA algorithm in our Algorithm 1.

Since our curves $Y^2 = X^5+uX^3+vX$ are in rather special form, they may be less likely suitable for cryptographic use. The author has no idea for theoretical results on this point. To observe the circumstance, the following numerical experiments are performed. Take a 87 bit number $A =$ 0x5072696d654e756d626572 in hexadecimal (the ASCII code for the string "PrimeNumber"). For the largest five primes p less than A satisfying $p \equiv 1 \bmod 4$ and $p \equiv 3 \bmod 4$, that is, for each prime $A-413$, $A-449$, $A-609$, $A-677$ and $A-957$ (they are 1 mod 4) and $A-23$, $A-35$, $A-39$, $A-179$, $A-267$ (they are 3 mod 4), Algorithm 1 is executed for 200 randomly generated $(u, v) \in \mathbf{F}_q \times (\mathbf{F}_q^{\times} - \mathbf{F}_q^4)$ (cf. Remark 2). In Table 1, we list the cofactors obtained in the numerical experiments. For each

curve, if the order of its Jacobian is a product of an integer less than 2^{10} and a prime, the value of the integer is listed in the second column. Thus the orders

Table 1. Distributions of cofactor returned by the algorithm

p	cofactors
$A-413$	2, 4, 4, 10, 10, 16, 16, 20, 40 80, 130, 208, 400, 580
$A-449$	2, 4, 4, 4, 26, 120, 394, 722, 740
$A-609$	2, 20, 52, 116, 256, 484, 820
$A-677$	4, 4, 8, 10, 20, 26, 34, 40, 82, 362, 400, 482, 740
$A-957$	2, 4, 4, 16, 20, 20, 20, 72, 90, 100, 262, 720
$A-23$	2, 2, 2, 14, 14, 16, 16, 34, 112, 184, 302
$A-35$	2, 8, 14, 18, 34, 56, 72, 194, 392
$A-39$	2, 2, 2, 8, 16, 62, 94, 98, 584, 656, 752, 784
$A-179$	2, 8, 18, 18, 32, 128, 146, 386, 512
$A-267$	2, 8, 8, 8, 14, 32, 34, 34, 50, 350, 446

of the Jacobians of those curves are divisible by primes greater than 2^{162}. In Table 2, we list values of u and v attained the best possible in the sense that the order of the Jacobian of the curve is two times of a prime: So, the probability to obtain the best possible curves in the above experiment is $\frac{13}{2000} = 0.0065$. By the prime number theorem, a "density" of primes around $A^2/2$ is approximately $1/\log_e(A^2/2) \approx 0.0084$. In the case $p = A - 677$, no best possible curve was found in the above computation. But further computation finds that the order of the Jacobian of the curve with $u = 935898796293578496671042 47$ and $v = 22435109140875626789358 13$ is a product of 2 and a prime

$$472920528303753106662785290707807991913732585053 4317.$$

This suggests that we can find a curve of the form $Y^2 = X^5 + uX^3 + vX$ for cryptographic use for any p. However, one might want to generate a random prime p and $u \in \mathbf{F}_p$, $v \in \mathbf{F}_p^\times$ for each curve.

Another observation is that curves with cofactor 6 and 12 were not found, where four curves with cofactor 18 were found. The author has no explanation of such phenomena.

The order of Jacobians of the curves $Y^2 = X^5 + uX^3 + vX$ seems less likely to be a product of a small positive integer and a large prime than a random integer. However, our experiments also suggests that our algorithm generates hyperelliptic curves with acceptable computational time.

We end this section with examples of curves to which we can apply Gaudry's efficient scalar multiplication [14]. Let $p = 2^{87} - 67$. Among 400 randomly generated pairs $(\alpha, \beta) \in \mathbf{F}_p^\times \times \mathbf{F}_p^\times$ such that $\alpha\beta$ is not a quadratic residue and that $\alpha \neq \pm\beta$, two of them gave the curve $Y^2 = X(X^2 - \alpha^2)(X^2 - \beta^2)$ with a

Table 2. Curves with cofactor 2

p	$u,\ v,$ order/2 (= prime)
$A-23$	26278410876831238768152256, 86364989829465111812877054, 472920528303659680345114257217571460043466532265912 7
$A-23$	48005263478608513799676536, 41220251281438002920145925, 472920528303659680345114258064366296670183406802 1159
$A-23$	20282462437998363621453892, 68366944575985333923275 45, 472920528303659680345114258133732817650284170540 2207
$A-35$	67648337171838833749692452, 56901612904748554441926559, 472920528303659680345114138072169120188545555476 7791
$A-39$	14846780454565145653845797, 21699720008937250121391434, 472920528303659680345114102353627309819453876693 2527
$A-39$	33146413473211029791343648, 80307184653756356179241 26, 472920528303659680345114100348567188574110420995 6991
$A-39$	71798113541184209350130597, 32655964372648182436520 42, 472920528303659680345114098177535158246221087181 2743
$A-179$	57049947468040934287481804, 79903956336370051333994749, 472920528303659680345112737905752263561298973395 1431
$A-267$	36056874937457171776037216, 25118608280476680778431722, 472920528303659680345111869267728944471081686145 071
$A-413$	83008004756000748681115585, 56563321965691537707290563, 472920528303561360104940130312551258067595321824 5917
$A-449$	28607196238761965671229454, 25170972284073882698894414, 472920528303673272123015906401321288517730330017 4073
$A-609$	95130756913811082727444951, 60322547687988038644971694, 472920528303771657998153663085486775700590453605 5113
$A-957$	10340770761867668646270127, 33230037969144840368622007, 472920528303608321884723156658609342418443268378 8437

cofactor 16. The corresponding values of u and v are

$$(u,v) = (1438092134922217781440405 47, 1311389691428426591040313 01),$$
$$(1520440429004836974725767 01, 4387381713480655712233614 6).$$

Since all 2-torsion points are $\mathbf{F}_q$-rational, the cofactor 16 is the best possible value. We need further experiments and analysis to observe the probability of such curves being suitable for cryptography.

8 Security considerations

We described an algorithm to generate a hyperelliptic curve $C/\mathbf{F}_q$ such that the order of Jacobian J of C is a product of a small number and a large prime. Since C is of the special form $Y^2 = X^5 + uX^3 + vX$, the curve C and its Jacobian J have some special structures. This might affect difficulties of the discrete log problems on $J(\mathbf{F}_q)$. We consider the problem.

First we note that the automorphism group of J is larger than that of a generic genus two hyperelliptic curve. One might think the larger automorphism group fasten the discrete log computation on $J(\mathbf{F}_p)$ by using the technique described in Duursma, Gaudry and Morain[7]. To some extent, it does. However, its influence is rather limited. Let us observe a reason. Let t be a time to perform one multiplication on $\mathbf{F}_p$. Let G be a (not necessarily finite, commutative) group acting on $J(\mathbf{F}_q)$ from left. We assume that the multiplication by (-1)-map belongs to G and that, expect for the (-1)-times map, a computation of the action of an element of G needs more time than t. Put $N = \#J(\mathbf{F}_q)$ and $\overline{N} = \#(G \setminus J(\mathbf{F}_q))$ for simplicity. The above mentioned method uses a random walk on the coset space $G \setminus J(\mathbf{F}_q)$ while original Pollard ρ method uses a random walk on $J(\mathbf{F}_q)$. Thus in order to solve DLP on $J(\mathbf{F}_q)$, the expected number of walks is reduced by a factor of $(N/\overline{N})^{1/2}$. However, we need to consider some extra costs. At each random walk step, we need to find a representative of a G-orbit to test a match on $G \setminus J(\mathbf{F}_q)$. For $P \in J(\mathbf{F}_q)$, the representative is a point having some properties (e.g. having a minimal X-coordinate regarded as an integer) among the points in the orbit $G \cdot P$. The time for this is no less than $t\#(G \cdot P)/2$. In average, this is $\frac{tN}{2\overline{N}}$. Let T be a time to compute the next point (not an equivalence class) by one step walk on $J(\mathbf{F}_q)$. Thus, the method in [7] runs faster than original Pollard ρ, in average, by a factor of

$$\frac{T\sqrt{N}}{\sqrt{\overline{N}}\left(T + \frac{tN}{2\overline{N}}\right)} \leq \sqrt{\frac{T}{2t}},$$

regardless of G. In our case, we may assume that $T/2t \leq 2^{16}$, for example. (In reality, we can perform an addition on $J(\mathbf{F}_q)$ with less than 100 multiplications over $\mathbf{F}_q$ and q is, at most, a small power of p.) We have only to increase the size of q by four bits.

A more powerful attack to our curve would be the index calculus method due to Gaudry[15]. In short, the method solve the DLP on an elliptic curve defined over $\mathbf{F}_{q^n}$ whose heuristic time complexity (with respect to the number of group operations) is $\tilde{O}(q^{2-(2/n)})$ as $q \to \infty$ for fixed n. The O-constant depends on n. Generically, a polynomial of degree $2^{n(n-1)}$ appears during the computation.

Now, it is obvious that the DLP on $J(\mathbf{F}_q)$ is reduced to that on $E/\mathbf{F}_q(s)$, where E and s are defined as in Section 3. Recall that $[\mathbf{F}_q(s) : \mathbf{F}_q]$ is either two or four. So, the time complexity of Gaudry's index calculus method is $\tilde{O}(q)$ and $\tilde{O}(q^{3/2})$, according to $n = 2$ and 4, respectively. On the other hand, the order of $J(\mathbf{F}_q)$ is $O(q^2)$ and square root algorithms solve the DLP on $J(\mathbf{F}_q)$ with $O(q)$ group operations. Thus, Gaurdy's index calculus method does not provide an asymptotically faster attack than square root algorithms, at least for the case $q = p$. In case of $q > p$, we can consider the Weil restriction of E to a subfield of $\mathbf{F}_q$ with a lager n. When $[\mathbf{F}_q : \mathbf{F}_p]$ is even and $[\mathbf{F}_q(s) : \mathbf{F}_q] = 2$, Gaudry's index calculus with $n = 4$ runs moderate space complexity and its time complexity is $\tilde{O}(q^{3/4})$, at least asymptotically. This case must be avoided. In the other cases, the efficiency of Gaudry's index calculus is questionable due to its huge space complexity. It is probably prudent to take $q = p$ in cryptographic applications.

Remark 7. We also need further tests for suitability to hyperelliptic curve cryptosystem for general hyperelliptic curves. These includes (but not limited to) the following conditions. The minimal embedding degree (in the sense of Hitt[21]) should not be too small to avoid multiplicative DLP reduction by Frey and Rück[10]. If M is close to $(\sqrt{q}-1)^2$ by some reason, the algorithm may return p as the largest prime factor. In this case, the input curve must not be used because it is vulnerable to the additive DLP reduction by Rück[34].

9 Conclusion

We presented an efficient algorithm to test whether a given hyperelliptic curve $Y^2 = X^5 + uX^3 + vX$ over $\mathbf{F}_q$ is suitable or not for cryptography and if suitable, the largest prime divisor of the number of $\mathbf{F}_q$-rational points of the Jacobian of the curve. This enables us to use a randomly generated hyperelliptic curve in the form which is supposed to be suitable for hyperelliptic curve cryptography. Although our family of curves is far more general than the curves given by binomials, it is still in a very special form. The difficulty of the discrete log problem on the simple but not absolutely simple Jacobian is open.

Acknowledgments

The major part of the work is performed during the author visited Prof. Steven Galbraith at Royal Holloway University. The author would like to thank their hospitality during his stay. He also would like to thank Steven Galbraith, Pierrick Gaudry, Florian Hess, Tanja Lange, Katsuyuki Takashima, Frederik Vercauteren and the anonymous reviewers for their helpful comments on this work.

References

1. Agrawal, M., Kayal, N., Saxena, N.: PRIMES is in P. Ann. of Math. 160, 781-793 (2004).
2. Anuradha, N.: Number of points on certain hyperelliptic curves defined over finite fields. Finite Fields Appl. 14, 314-328 (2008). doi: 10.1016/j.ffa.2006.12.007
3. Berndt, B.C., Evans, R.J., Williams, K.S.: Gauss and Jacobi sums. John Wiley & Sons, Inc, New York. (1998).
4. Cantor, D., Zassenhaus, H.: A new algorithm for factoring polynomials over finite fields. Math. Comp. 36, 587-592 (1981).
5. Cassels, J.W.S., Flynn, E.V.: Prolegomena to a middlebrow arithmetic of curves of genus 2. London Math. Soc. Lecture Note Series, vol. 230. Cambridge Univ. Press, Cambridge. (1996).
6. Chebyshev, P.L.: Mémoire sur les nombres premiers. J. Math. Pures Appl. 17, 366-390 (1852), Œuvres, I-5.
7. Duursma, I., Gaudry, P., Morain, F.: Speeding up the discrete log computation on curves with automorphisms. In: Advances in cryptology - Asiacrypt'99, vol. 1716, pp. 103-121. Springer, Berlin, Heidelbert(1999). doi: 10.1007/b72231

8. Elkies, N.D.: Elliptic and modular curves over finite fields and related computational issues. In: Computational perspectives on number theory (Chicago, IL, 1995), AMS/IP Stud. Adv. Math., vol. 7, pp. 21-76. AMS, Providence, RI(1998).
9. Frey, G., Kani, E.: Curves of genus 2 covering elliptic curves and an arithmetical application. In: van der Geer, G., Oort, F., Steenbrink, J. (ed.) Arithmetic algebraic geometry (Texel, 1989), Progress in Math., vol. 89, pp. 153-176. Birkhäuser Boston, Boston(1991).
10. Frey, G., Rück, H.-G.: A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. Math. Comp. 62, 865-874 (1994).
11. Furukawa, E., Kawazoe, M., Takahashi, T.: Counting points for hyperelliptic curves of type $y^2 = x^5 + ax$ over finite prime fields. In: Selected areas in cryptography, 2003, Lect. Notes in Comput. Sci., vol. 3006, pp. 26-41. Springer, Berlin, Heiderberg(2004). doi: 10.1007/b96837
12. Gallant, R., Lambert, R., Vanstone, S.: Faster point multiplication on elliptic curves with efficient endomorphisms. In: Kilian, J. (ed.) Advances in cryptology - Proceedings of CRYPTO 2001, Lect. Notes in Comput. Sci., vol. 2139, pp. 190-200. Springer, Berlin(2001). doi: 10.1007/3-540-44647-8_11
13. Gaudry, P.: An algorithm for solving the discrete log problem on hyperellipitc curves. In: Preneel, B. (ed.) Advances in cryptology – Eurocrypt 2000, Lect. Notes in Comput. Sci., vol. 1807, pp. 19-34. Springer, Berlin(2000). doi: 10.1007/3-540-45539-6_2
14. Gaudry, P.: Fast genus 2 arithmetic based on Theta functions. J. Math. Cryptology 1, 243-265 (2007). doi: 10.1515/JMC.2007.012
15. Gaudry, P.: Index calculus for abelian varieties of small dimension and the elliptic curve discrete logarithm problem. J. Symbolic Comput. (2008), to appear. doi: 10.1016/j.jsc.2008.08.005
16. Gaudry, P., Schost, É.: On the invariants of the quotients of the Jacobian of a curve of genus 2. In: Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, AAECC-14 (Melborune, 2001), Lect. Notes in Comput. Sci., vol. 2227, pp. 373-386. Springer, Berlin, Heidelberg(2001). doi: 10.1007/3-540-45624-4_39
17. Gaudry, P., Schost, É.: Construction of secure random curves of genus 2 over prime fields. In: Cachin, C., Camenisch, J. (ed.) Advances in cryptology - EUROCRYPT 2004, Lect. Notes in Comput. Sci., vol. 3027, pp. 239-256. Springer, Berlin, Heidelberg(2004). doi: 10.1007/b97182
18. Gaudry, P., Schost, É.: *Hyperelliptic point counting record: 254 bit Jacobian.* (2008) Post to NMBRTHRY list, 22 Jun 2008..
19. Haneda, M., Kawazoe, M., Takahashi, T.: Suitable curves for genus-4 HCC over prime fields: point counting formulae for hyperelliptic curves of type $y^2 = x^{2k+1} + ax$. In: Automata, Languages and Programming, 32nd International Colloquium, ICALP 2005, Lect. Notes in Comput. Sci., vol. 3580, pp. 539-550.(2005). doi: 10.1007/11523468_44
20. Hess, F., Seroussi, G., Smart, N.P.: Two Topics in Hyperelliptic Cryptography. In: Selected areas in cryptography, SAC, Lect. Notes in Comput. Sci., vol. 2259, pp. 181-189. Springer, Berlin, Heiderberg(2001). doi: 10.1007/3-540-45537-X_14
21. Hitt, L.: On the minimal embedding field. In: Pairing-based cryptography – Pairing 2007, Lect. Notes in Comput. Sci., vol. 4575, pp. 294-301. Springer, Berlin, Heidelberg(2007). doi: 10.1007/978-3-540-73489-5_16
22. Kedlaya, K.: Counting points on hyperelliptic curves using Monsky-Washnitzer cohomology. J. Ramanujan Math. Soc. 16, 323-338 (2001).
23. Lauder, A.G.B.: Rigid cohomology and p-adic point counting. J. Théor. Nombres Bordeaux 17, 169-180 (2005).

24. Leprévost, F., Morain, F.: Revêtements de courbes elliptiques à multiplication complexe par des courbes hyperelliptiques et sommes de caractéres. J. Number Theory 64, 165-182 (1997). doi: 10.1006/jnth.1997.2070
25. Lercier, R.: Finding Good Random Elliptic Curves for Cryptosystems Defined over $\mathbf{F}_{2^n}$. In: Fumy, W. (ed.) Advances in Cryptology - EUROCRYPT'97 (Konstanz), Lect. Notes in Comput. Sci., vol. 1233, pp. 379-392. Springer, Berlin, Heidelberg(1997). doi: 10.1007/3-540-69053-0_26
26. Lercier, R., Lubicz, D.: A quasi quadratic time algorithm for hyperelliptic curve point counting. Ramanujan J. 12, 399-423 (2006). doi: 10.1007/s11139-006-0151-6
27. Matsuo, K., Chao, J., Tsujii, S.: An improved baby step giant step algorithm for point counting of hyperelliptic curves over finite fields. In: Fieker, C., Kohel, D. (ed.) Algorithmic number theory (Sydney, Australia, July 2002), Lect. Notes in Comput. Sci., vol. 2369, pp. 461-474. Springer, Berlin(2002). doi: 10.1007/3-540-45455-1_36
28. Maurer, M., Müller, V.: Finding the eigenvalue in Elkies' algorithm. Experimental Math. 10, 275-285 (2001).
29. Milne, J.S.: Abelian varieties. In: Cornell, G., Silverman, J.H. (ed.) Arithmetic Geometry, pp. 103-150. Springer, New York(1986).
30. Milne, J.S.: Jacobian varieties. In: Cornell, G., Silverman, J.H. (ed.) Arithmetic Geometry, pp. 167-212. Springer, New York(1986).
31. Paulhus, J.: Decomposing Jacobians of curves with extra automorphisms. Acta Arith. 132, 231-244 (2008).
32. Pila, J.: Frobenius maps of Abelian varieties and finding roots of unity in finite fields. Math. Comp. 55, 745-763 (1990).
33. Pohlig, S. C., Hellman, M. E.: An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance. IEEE Trans. Info. Theory 24, 106-110 (1978).
34. Rück, H. G.: On the discrete logarithm in the divisor class group of curves. Math. Comp. 68, 805-806 (1999). doi: 10.1090/S0025-5718-99-01043-1
35. Scholten, J.: *Weil restriction of an elliptic curve over a quadratic extension.* preprint, available at http://homes.esat.kuleuven.be/~jscholte/.
36. Schoof, R.: Elliptic curves over finite fields and the computation of square roots mod p. Math. Comp. 44, 483-494 (1985).
37. Schoof, R.: Counting points on elliptic curves over finite fields. J. Théor. Nombres Bordeaux 7, 219-254 (1995).
38. Sutherland, A.V.: A generic apporach to searching for Jacobians. Math. Comp. 78, 485-507 (2009). doi: 10.1090/S0025-5718-08-02143-1
39. Takashima, K.: A new type of fast endomorphisms on Jacobians of hyperelliptic curves and their cryptographic application. IEICE Trans. Fundamentals E89-A, 124-133 (2006).
40. Vercauteren, F.: Advances in point counting. In: Blake, I.F., Seroussi, G., Smart, N.P. (ed.) Advances in elliptic curve cryptography, London Math. Sco. Lecture Note Ser., vol. 317, pp. 103-132. Cambridge Univ. Press, Cambridge(2005).
41. Vercautern, F.: *The SEA algorithm in characteristic 2.* (2000) preprint, available at http://homes.esat.kuleuven.be/~fvercaut/papers/SEA.pdf.
42. von zur Gathen, J., Gerhard, J.: Modern computer algebra (2nd ed.). Cambridge UP, Cambridge. (2003).
43. von zur Gathen, J., Shoup, V.: Computing Frobenius maps and factoring polynomials. Computational complexity 2, 187-224 (1992). doi: 10.1007/BF01272074