LETTER

Skew-Frobenius Maps on Hyperelliptic Curves

Shunji KOZAKI[†a)], *Nonmember*, **Kazuto MATSUO**[†], *Member*, *and* **Yasutomo SHIMBARA**[†], *Nonmember*

SUMMARY Scalar multiplication methods using the Frobenius maps are known for efficient methods to speed up (hyper)elliptic curve cryptosystems. However, those methods are not efficient for the cryptosystems constructed on fields of small extension degrees due to costs of the field operations. Iijima et al. showed that one can use certain automorphisms on the quadratic twists of elliptic curves for fast scalar multiplications without the drawback of the Frobenius maps. This paper shows an extension of the automorphisms on the Jacobians of hyperelliptic curves of arbitrary genus.

key words: *hyperelliptic curve cryptosystems, scalar multiplications, Frobenius expansions, skew-Frobenius maps*

1. Introduction

Hyperelliptic curve cryptosystems [9] are public-key cryptosystems whose keys are as short as those of elliptic curve cryptosystems. Furthermore, the definition fields of hyperelliptic curve cryptosystems can be more compact than those of elliptic curve cryptosystems. Choosing a compact field suitable for a general-purpose processor achieves fast group operations on the Jacobian of a hyperelliptic curve [6], [12].

In the implementation of hyperelliptic curve cryptosystems, fast scalar multiplication methods are important. The scalar multiplication method using a subfield curve [2, Sect. 15.1] is one of the fastest methods. The method uses the q-th power Frobenius map for $\mathbb{F}_{q^n}$-rational points on the Jacobian of a curve defined over $\mathbb{F}_q$.

However, there is a drawback of the method, i.e., the order of $\mathbb{F}_{q^n}$-rational point group is always composite, because the group contains $\mathbb{F}_q$-rational points as a subgroup. Hence, $\mathbb{F}_{q^n}$ is larger than the field defining a rational point group of prime order with the same security level. Especially, those methods are inefficient when the extension degree n is small.

Iijima, Matsuo, Chao, and Tsujii [7] showed that, on the quadratic twists of elliptic curves, one can obtain automorphisms which have the similar properties to the Frobenius maps. We call these kinds of automorphisms skew-Frobenius maps. The skew-Frobenius maps are efficiently computable and can be used to speed up scalar multiplications.

This paper extends the skew-Frobenius maps in [7] to hyperelliptic curves of any genus. The skew-Frobenius map is obtained as a computable map on the Jacobian of the quadratic twist of a hyperelliptic curve.

The organization of this paper is as follows: Sect. 2 defines the quadratic twists of hyperelliptic curves, the Frobenius maps on the Jacobians and the related facts. Section 3 presents a brief overview of the skew-Frobenius maps on elliptic curves proposed by [7]. Section 4 constructs the skew-Frobenius maps for hyperelliptic curves. Moreover, Sect. 5 shows the complexity of the maps, Sect. 6 briefly discusses the efficiency of the skew-Frobenius maps for scalar multiplications. Finally, Sect. 7 concludes this paper.

2. Preliminaries

Let p be an odd prime, m a positive integer and $q = p^m$. Let g be a positive integer and F a monic square-free polynomial in $\mathbb{F}_q[X]$ of degree $2g + 1$. A hyperelliptic curve C over $\mathbb{F}_q$ of genus g is defined by

$$C : Y^2 = F(X). \tag{1}$$

Let n be an integer greater than 1 and c a quadratic non-residue in $\mathbb{F}_{q^n}$. The quadratic twist C_t of C over $\mathbb{F}_{q^n}$ is defined by

$$C_t : \ Y^2 = F_t(X), \tag{2}$$

$$F_t(X) = c^{2g+1}F(c^{-1}X) \in \mathbb{F}_{q^n}[X]. \tag{3}$$

$\mathbb{J}_C$ denotes the Jacobian of C and $\overline{\mathbb{F}}_q$ denotes the algebraic closure of $\mathbb{F}_q$. Any element $\mathcal{D} \in \mathbb{J}_C$ can be uniquely represented by a pair of polynomials U and V in $\overline{\mathbb{F}}_q[X]$ which satisfy the following conditions [2, Theorem 14.5]:

$$U \text{ is monic}, \tag{4}$$

$$\deg V < \deg U \le g, \tag{5}$$

$$F - V^2 \equiv 0 \pmod{U}. \tag{6}$$

The pair (U, V) is called Mumford representation of $\mathcal{D}$ and we write $\mathcal{D} = (U, V)$. An addition of two elements in $\mathbb{J}_C$ can be computed by Cantor's algorithm (Algorithm 1) [1].

The Frobenius map ϕ_q on $\mathbb{J}_C$ is defined by

$$\begin{aligned} &\phi_q : \mathbb{J}_C \to \mathbb{J}_C \\ &(U, V) \mapsto (\phi_q(U), \phi_q(V)), \end{aligned} \tag{7}$$

where $\phi_q(U)$ (resp., $\phi_q(V)$) is the polynomial obtained by raising each coefficient of U (resp., V) to the q-th power. Every element in $\mathbb{J}_C$ is defined over $\mathbb{F}_{q^n}$ if and only if it is

Manuscript received December 25, 2007.
Manuscript revised March 24, 2008.

[†]The authors are with the Institute of Information Security, Yokohama-shi, 221-0835 Japan.
a) E-mail: dgs074103@iisec.ac.jp

DOI: 10.1093/ietfec/e91–a.7.1839

Copyright © 2008 The Institute of Electronics, Information and Communication Engineers

Algorithm 1 Cantor's algorithm

Input: C: a hyperelliptic curve of genus g over $\mathbb{F}_q$,
$\mathcal{D}_1 = (U_1, V_1), \mathcal{D}_2 = (U_2, V_2) \in \mathbb{J}_C$
Output: $\mathcal{D} = (U_3, V_3) \in \mathbb{J}_C$ such that $\mathcal{D} = \mathcal{D}_1 + \mathcal{D}_2$
/*composition part*/
1: Compute $G, S_1, S_2, S_3 \in \overline{\mathbb{F}}_q[X]$ such that
$G = \gcd(U_1, U_2, V_1 + V_2) = S_1U_1 + S_2U_2 + S_3(V_1 + V_2)$
2: $U_3 \leftarrow \frac{U_1U_2}{G^2}$
3: $V_3 \leftarrow \frac{S_1U_1V_2 + S_2U_2V_1 + S_3(F + V_1V_2)}{G} \bmod U_3$
/*reduction part*/
4: **repeat**
5: $U_3 \leftarrow \frac{F - V_3^2}{U_3}$
6: $V_3 \leftarrow -V_3 \bmod U_3$
7: **until** $\deg U_3 \le g$
8: Make U_3 monic
9: **return** (U_3, V_3)

fixed by ϕ_q^n. The set of the elements defined over $\mathbb{F}_{q^n}$ is denoted by $\mathbb{J}_C(\mathbb{F}_{q^n})$. $\mathbb{J}_C(\mathbb{F}_{q^n})$ is a finite abelian subgroup of $\mathbb{J}_C$.

The characteristic polynomial of ϕ_q is of the form

$$\chi_q(X) = X^{2g} + \sum_{i=1}^{g} a_i X^{2g-i} + \sum_{i=1}^{g-1} a_i q^{g-i} X^i + q^g \in \mathbb{Z}[X],\ a_i \in \mathbb{Z},\ 1 \le i \le g$$

[2, Corollary 5.82]. By using $\chi_q(\phi_q) = 0$, one can expand an integer k in $\mathbb{Z}[\phi_q]$ as follows [2, Sect. 15.1.2]:

$$k = \sum_{i=0}^{s} r_i \phi_q^i,\ r_i \in \mathbb{Z},\ s \in \mathbb{N}. \tag{8}$$

Since the map ϕ_q is more efficiently computable than the group operation of $\mathbb{J}_C(\mathbb{F}_{q^n})$, the ϕ_q-expansion (8) leads to fast scalar multiplications.

In order to use the ϕ_q-expansions, however, we must build a cryptosystem on $\mathbb{J}_C(\mathbb{F}_{q^n})/\mathbb{J}_C(\mathbb{F}_q)$. Therefore, the definition field $\mathbb{F}_{q^n}$ is larger than that of a rational point group of prime order with the same security level. Since $\sharp(\mathbb{J}_C(\mathbb{F}_{q^n})/\mathbb{J}_C(\mathbb{F}_q)) \approx q^{g(n-1)}$ from the Hasse-Weil theorem [2, Theorem 14.15], the methods using the ϕ_q-expansions are not efficient when n is small.

3. Skew-Frobenius Maps on Elliptic Curves [7]

Iijima, Matsuo, Chao, and Tsujii [7] showed the skew-Frobenius maps on the quadratic twists of elliptic curves. Those maps can be used for expanding integers in the endomorphism rings of curves similar to the Frobenius map. [7] also showed that there are rational point groups of prime order on which the skew-Frobenius maps can be used. This section recalls the skew-Frobenius maps on elliptic curves proposed in [7].

Let E be an elliptic curve over $\mathbb{F}_q$ defined by (1) with $g = 1$ and E_t the quadratic twist of E defined by (2) using a quadratic non-residue c in $\mathbb{F}_{q^n}$. A map

$$\begin{aligned} \tau_0 : E_t &\to E \\ (x, y) &\mapsto (c^{-1}x, c^{-\frac{3}{2}}y) \end{aligned} \tag{9}$$

defines an isomorphism from E_t to E. The skew-Frobenius map on E_t is defined by $\tau_0^{-1} \circ \phi_q \circ \tau_0$, i.e.,

$$\begin{aligned} \phi_t : E_t &\to E_t \\ (x, y) &\mapsto (c^{1-q}x^q, c^{\frac{3}{2}(1-q)}y^q). \end{aligned}$$

The restriction of ϕ_t to the $\mathbb{F}_{q^n}$-rational point group $E_t(\mathbb{F}_{q^n})$ is also a non-trivial automorphism which satisfies $\chi_q(\phi_t) = 0$. Then [7] showed that there is the quadratic twist E_t of E such that $\sharp E_t(\mathbb{F}_{q^n})$ is a prime when $n = 2^d, d > 0$. Moreover, Furukawa, Kobayashi, and Aoki [5] showed the finite field representations suitable for the skew-Frobenius maps. Furihata, Kobayashi, and Saito [4] showed the skew-Frobenius maps on elliptic curves over fields of characteristics two.

4. Skew-Frobenius Maps on Hyperelliptic Curves

This section shows the skew-Frobenius map on $\mathbb{J}_{C_t}$ of the quadratic twist C_t of a hyperelliptic curve C of any genus.

Lemma 1 described below gives the isomorphism from $\mathbb{J}_{C_t}$ to $\mathbb{J}_C$. This map is an extension of the map (9).

Lemma 1.

$$\begin{aligned} \tau : \mathbb{J}_{C_t} &\to \mathbb{J}_C \\ (U, V) &\mapsto (\tilde{U}, \tilde{V}) \end{aligned} \tag{10}$$

$$\tilde{U}(X) = c^{-\ell} U(cX), \ell = \deg U \tag{11}$$

$$\tilde{V}(X) = c^{-\frac{2g+1}{2}} V(cX) \tag{12}$$

is an isomorphism from $\mathbb{J}_{C_t}$ to $\mathbb{J}_C$, where c is the quadratic non-residue in $\mathbb{F}_{q^n}$ that appears in (3).

Proof. First, we show that τ is a well defined map from $\mathbb{J}_{C_t}$ to $\mathbb{J}_C$, that is, $\mathbb{J}_C$ contains $(\tilde{U}, \tilde{V})$ given by (11) and (12) for any $(U, V) \in \mathbb{J}_{C_t}$. Obviously, $\tilde{U}$ satisfies the condition (4). Since $\deg V < \deg U \le g$,

$$\deg \tilde{V} = \deg V < \deg \tilde{U} = \deg U \le g.$$

Therefore, $(\tilde{U}, \tilde{V})$ satisfies the condition (5). From $F(X) = c^{-(2g+1)} F_t(cX)$ and (12),

$$F(X) - \tilde{V}^2(X) = c^{-(2g+1)} \left(F_t(cX) - V^2(cX) \right).$$

Then $F_t(cX) - V^2(cX) \equiv 0 \pmod{U(cX)}$ implies

$$F(X) - \tilde{V}^2(X) \equiv 0 \pmod{\tilde{U}(X)},$$

that is, $(\tilde{U}, \tilde{V})$ satisfies the condition (6). Consequently, $(\tilde{U}, \tilde{V})$ is contained in $\mathbb{J}_C$.

Next, we show that τ is a group homomorphism, that is, $\tau(\mathcal{D}_1 + \mathcal{D}_2)$ is equal to $\tau(\mathcal{D}_1) + \tau(\mathcal{D}_2)$ for any $\mathcal{D}_1, \mathcal{D}_2 \in \mathbb{J}_{C_t}$ by using Algorithm 1. Let $(\tilde{U}_i, \tilde{V}_i) := \tau(\mathcal{D}_i)$ for $\mathcal{D}_i = (U_i, V_i) \in \mathbb{J}_{C_t}$, $i = 1, 2$. From (11) and (12),

$$\tilde{U}_i(X) = c^{-\ell_i} U_i(cX),\ \ell_i = \deg U_i \tag{13}$$

$$\tilde{V}_i(X) = c^{-\frac{2g+1}{2}} V_i(cX) \tag{14}$$

for $i = 1, 2$. According to the step 1 in Algorithm 1, let $G := \gcd(U_1, U_2, V_1 + V_2)$ and $\tilde{G} := \gcd(\tilde{U}_1, \tilde{U}_2, \tilde{V}_1 + \tilde{V}_2)$, where both G and $\tilde{G}$ are monic. By using (13) and (14), the following is obtained:

$$\begin{aligned}\tilde{G} &= \gcd(U_1(cX), U_2(cX), V_1(cX) + V_2(cX)) \\ &= c^{-\ell_G} G(cX),\ \ell_G = \deg G. \end{aligned} \tag{15}$$

Moreover, let $S_1, S_2, S_3 \in \overline{\mathbb{F}}_{q^n}[X]$ (resp., $\tilde{S_1}, \tilde{S_2}, \tilde{S_3} \in \overline{\mathbb{F}}_{q^n}[X]$) such that $G = S_1U_1 + S_2U_2 + S_3(V_1 + V_2)$ (resp., $\tilde{G} = \tilde{S_1}\tilde{U_1} + \tilde{S_2}\tilde{U_2} + \tilde{S_3}(\tilde{V_1} + \tilde{V_2})$). Then, applying (13) and (14) to (15), we obtain

$$\begin{aligned}\tilde{G} &= c^{-\ell_G}(S_1(cX)U_1(cX) + S_2(cX)U_2(cX) \\ &\quad + S_3(cX)(V_1(cX) + V_2(cX))) \\ &= c^{\ell_1-\ell_G} S_1(cX)\tilde{U_1}(X) + c^{\ell_2-\ell_G} S_2(cX)\tilde{U_2}(X) \\ &\quad + c^{\frac{2g+1}{2}-\ell_G} S_3(cX)(\tilde{V_1}(X) + \tilde{V_2}(X)).\end{aligned}$$

Therefore, $\tilde{S_1}$, $\tilde{S_2}$ and $\tilde{S_3}$ satisfy

$$\tilde{S_i} = c^{\ell_i-\ell_G} S_i(cX),\ \ell_i = \deg S_i,\ i = 1, 2, \tag{16}$$

$$\tilde{S_3} = c^{\frac{2g+1}{2}-\ell_G} S_3(cX). \tag{17}$$

According to the step 2 in the algorithm, let

$$\begin{aligned}U_3 &:= \frac{U_1U_2}{G^2},\ \ell_3 = \deg U_3 = \ell_1 + \ell_2 - 2\ell_G, \\ \tilde{U_3} &:= \frac{\tilde{U_1}\tilde{U_2}}{\tilde{G}^2},\end{aligned}$$

then applying (13) and (15) to $\tilde{U_3}$ yields

$$\tilde{U_3} = c^{-\ell_3} U_3(cX). \tag{18}$$

According to the step 3 in the algorithm, let

$$\begin{aligned}V_3 &:= \frac{S_1U_1V_2 + S_2U_2V_1 + S_3(F_t + V_1V_2)}{G} \bmod U_3, \\ \tilde{V_3} &:= \frac{\tilde{S_1}\tilde{U_1}\tilde{V_2} + \tilde{S_2}\tilde{U_2}\tilde{V_1} + \tilde{S_3}(F + \tilde{V_1}\tilde{V_2})}{\tilde{G}} \bmod \tilde{U_3},\end{aligned}$$

then applying (13), (14), (15), (16), (17) and (18) to $\tilde{V_3}$ yields

$$\tilde{V_3} = c^{-\frac{2g+1}{2}} V_3(cX). \tag{19}$$

According to the step 5 in the algorithm, let

$$\begin{aligned}U_3' &:= \frac{F_t - V_3^2}{U_3},\ \ell_3' = \deg U_3' = 2g + 1 - \ell_3, \\ \tilde{U_3}' &:= \frac{F - \tilde{V_3}^2}{\tilde{U_3}},\end{aligned}$$

then by using (18), (19) and $F(X) = c^{-(2g+1)} F_t(cX)$,

$$\tilde{U_3}' = c^{-\ell_3'} U_3'(cX) \tag{20}$$

is obtained. According to the step 6 in the algorithm, let

$$\begin{aligned}V_3' &:= -V_3 \bmod U_3', \\ \tilde{V_3}' &:= -\tilde{V_3} \bmod \tilde{U_3}',\end{aligned}$$

then by using (19) and (20),

$$\tilde{V_3}' = c^{-\frac{2g+1}{2}} V_3'(cX) \tag{21}$$

is obtained. Since (20) and (21) are always satisfied between the step 4 and 7 in the algorithm, $\mathcal{D}_1 + \mathcal{D}_2 = (U, V)$ implies

$$\tau(\mathcal{D}_1) + \tau(\mathcal{D}_2) = \left(c^{-\ell} U(cX), c^{-\frac{2g+1}{2}} V(cX)\right),$$

where $\ell := \deg U \le g$. Therefore,

$$\tau(\mathcal{D}_1) + \tau(\mathcal{D}_2) = \tau(\mathcal{D}_1 + \mathcal{D}_2),$$

that is, τ is a group homomorphism.

Similarly, we can show that a map

$$\begin{aligned}&\sigma : \mathbb{J}_C \to \mathbb{J}_{C_t} \\ &(U, V) \mapsto \left(c^{\ell} U(c^{-1}X), c^{\frac{2g+1}{2}} V(c^{-1}X)\right), \\ &\qquad \ell = \deg U\end{aligned} \tag{22}$$

is a group homomorphism. Moreover, $\tau \circ \sigma = \mathrm{id}_{\mathbb{J}_C}$, $\sigma \circ \tau = \mathrm{id}_{\mathbb{J}_{C_t}}$ are satisfied. Therefore, τ is an isomorphism from $\mathbb{J}_{C_t}$ to $\mathbb{J}_C$. □

The Frobenius map ϕ_q on $\mathbb{J}_C$ and the isomorphism τ in Lemma 1 provide an automorphism on $\mathbb{J}_{C_t}$ as follows.

Definition 1. *The skew-Frobenius map $\tilde{\phi}_q$ on $\mathbb{J}_{C_t}$ is defined by*

$$\tilde{\phi}_q : \mathbb{J}_{C_t} \xrightarrow[\tau]{\sim} \mathbb{J}_C \xrightarrow[\phi_q]{\sim} \mathbb{J}_C \xrightarrow[\tau^{-1}]{\sim} \mathbb{J}_{C_t}.$$

From the definition, $\tilde{\phi}_q$ satisfies $\chi_q(\tilde{\phi}_q) = 0$. Theorem 1 described below shows an explicit map of $\tilde{\phi}_q$ with respect to the Mumford representation.

Theorem 1. *The skew-Frobenius map $\tilde{\phi}_q$ on $\mathbb{J}_{C_t}$ is given by*

$$\begin{aligned}&\tilde{\phi}_q : \mathbb{J}_{C_t} \to \mathbb{J}_{C_t} \\ &(U, V) \mapsto (\bar{U}, \bar{V}), \\ &\bar{U}(X) = X^{\ell} + \sum_{i=0}^{\ell-1} c^{(1-q)(\ell-i)} u_i^q X^i, \\ &\bar{V}(X) = \sum_{i=0}^{k} c^{(1-q)\left(\frac{2g+1}{2}-i\right)} v_i^q X^i,\end{aligned} \tag{23}$$

where

$$U(X) = X^{\ell} + \sum_{i=0}^{\ell-1} u_i X^i \in \overline{\mathbb{F}}_{q^n}[X],$$

$$V(X) = \sum_{i=0}^{k} v_i X^i \in \overline{\mathbb{F}}_{q^n}[X].$$

Proof. From (11), (7) and (22),

$$\begin{aligned}\bar{U}(X) &= c^{-(q-1)\ell}\phi_q(U)(c^{q-1}X)\\ &= X^\ell + \sum_{i=0}^{\ell-1} c^{(1-q)(\ell-i)}u_i^q X^i\end{aligned}$$

is obtained. From (12), (7) and (22),

$$\begin{aligned}\bar{V}(X) &= c^{-(q-1)\frac{2g+1}{2}}\phi_q(V)(c^{q-1}X)\\ &= \sum_{i=0}^{k} c^{(1-q)\left(\frac{2g+1}{2}-i\right)}v_i^q X^i\end{aligned}$$

is obtained. □

From Theorem 1, the restriction of $\tilde{\phi}_q$ to $\mathbb{J}_{C_t}(\mathbb{F}_{q^n})$ is also a non-trivial automorphism on $\mathbb{J}_{C_t}(\mathbb{F}_{q^n})$, because $c^{(1-q)\left(\frac{2g+1}{2}-i\right)}$ in (23) is in $\mathbb{F}_{q^n}$ for any i. Therefore, one can use $\tilde{\phi}_q$ to expand integers in $\mathbb{Z}[\tilde{\phi}_q]$ for scalar multiplications on $\mathbb{J}_{C_t}(\mathbb{F}_{q^n})$.

5. Complexity of Skew-Frobenius Map

This section discusses the complexity of computing $\tilde{\phi}_q$ with respect to g.

From Theorem 1, the precomputation of

$$\{c^{(1-q)}, c^{\frac{3}{2}(1-q)}, c^{2(1-q)}, \ldots, c^{g(1-q)}, c^{\frac{2g+1}{2}(1-q)}\}$$

allows us to compute $\tilde{\phi}_q$ by using at most $2g$ $\mathbb{F}_{q^n}$-multiplications and at most $2g$ q-th powers. Therefore, $\tilde{\phi}_q$ can be computed using $O(g)$ operations in $\mathbb{F}_{q^n}$. On the other hand, a group operation on $\mathbb{J}_{C_t}(\mathbb{F}_{q^n})$ needs $O(g^2)$ (or $O(g^3)$) operations in $\mathbb{F}_{q^n}$. Comparing the cost of $\tilde{\phi}_q$ computation with that of the group operation with respect to g, the increase in the cost of $\tilde{\phi}_q$ computation with g is smaller than that in the cost of the group operation. Moreover, $\tilde{\phi}_q$ allows us to use $\mathbb{J}_{C_t}(\mathbb{F}_{q^n})$ of prime order [8], so that we can reduce the cost of operations in $\mathbb{F}_{q^n}$ compared to using a rational point group of composite order with the same security level.

6. Efficiency of Skew-Frobenius Maps for Scalar Multiplications

Using the results in [11] and [3], this section briefly discusses the efficiency of the skew-Frobenius maps for scalar multiplications.

$\tilde{\phi}_q$ has the same characteristic polynomial as ϕ_q. Therefore, according to [11, Algorighm 1], we can obtain the skew-Frobenius expansion for an integer $k \approx q^{gn}$ as

$$k = \sum_{i=0}^{\lambda-1} a_i\tilde{\phi}_q^i,\ a_i \in \left\{0, \pm 1, \ldots, \pm\frac{q^g-1}{2}\right\}. \tag{24}$$

Most of these expansions have finite length due to [11, Sect. 4.2, 4.3], and [11, Summary 1] shows that the upper bound of λ is

$$\left\lceil 2\log_q(2(\sqrt{q}-1)k)\right\rceil + k_{q,g},$$

where $k_{q,g} \le 2g+1$ as an experimental result in [11]. From $\log_q 2(\sqrt{q}-1) < 1$ and $\log_q k < ng+1$, we see that

$$\begin{aligned}&\left\lceil 2\log_q(2(\sqrt{q}-1)k)\right\rceil + k_{q,g}\\ &\quad\le 2\left\lceil \log_q(2(\sqrt{q}-1)k)\right\rceil + 2g + 1\\ &\quad< 2\left\lceil 1+\log_q k\right\rceil + 2g + 1 = 2\left\lceil \log_q k\right\rceil + 2g + 3\\ &\quad< 2(ng+1) + 2g + 3 = 2g(n+1) + 5.\end{aligned}$$

Thus, the upper bound of λ is $2g(n+1)+4$. Moreover, $\tilde{\phi}_q$ satisfies $\tilde{\phi}_q^n(\mathcal{D}) = -\mathcal{D}$ for any $\mathcal{D} \in \mathbb{J}_{C_t}(\mathbb{F}_{q^n})$, i.e. $\tilde{\phi}_q^n = -1$. Therefore, the expansion (24) can be rewritten with the terms of degree less than n:

$$k = \sum_{i=0}^{L} r_i\tilde{\phi}_q^i,\ \log_2|r_i| \le M, \tag{25}$$

where

$$\begin{aligned}&L = n-1,\\ &M = \log_2\left(\left(\frac{q^g-1}{2}\right)\left\lceil\frac{2g(n+1)+4}{n}\right\rceil\right).\end{aligned}$$

According to [3, Sect. 4], we can compute $k\mathcal{D}$ by using the expansion (25) as follows.

$$\begin{aligned}k\mathcal{D} &= 2\left(\cdots 2\left(\sum_{i=0}^{L} r_i^{(M-1)}\tilde{\phi}_q^i(\mathcal{D})\right) + \cdots\right)\\ &\quad + \sum_{i=0}^{L} r_i^{(0)}\tilde{\phi}_q^i(\mathcal{D}),\end{aligned}$$

where $(r_i^{(M-1)}, \ldots, r_i^{(0)})_2$ is the binary representation of r_i for $0 \le i \le L$. Consequently, $k\mathcal{D}$ can be computed by using $\frac{LM}{2} + M - 1$ additions on average and $M-1$ doublings.

For example, taking a typical case, let $g=2$, $q \approx 2^{20}$, and $n=4$, then the cost of a scalar multiplication using the skew-Frobenius map can be estimated as follows. We can consider the cost of additions and doublings are roughly the same, then the computation of $k\mathcal{D}$ needs $\left(\frac{L}{2}+2\right)M-2$ group operations. Therefore, $k\mathcal{D}$ can be computed within

$$\begin{aligned}&\left(\frac{L}{2}+2\right)M - 2\\ &\quad= \frac{n+3}{2}\log_2\left(\left(\frac{q^g-1}{2}\right)\left\lceil\frac{2g(n+1)+4}{n}\right\rceil\right) - 2\\ &\quad\approx 144\end{aligned}$$

group operations. On the other hand, the binary method [2, Sect. 9.1.1] needs about 238 group operations with the same parameters. Therefore, the scalar multiplication using the skew-Frobenius map reduces about 39% of the group operations compared with the binary method. Moreover, the NAF_w method [2, Sect. 9.1.4] needs about 193 group operations when the window size $w=5$. Therefore, the scalar multiplication using the skew-Frobenius map reduces about 25% of the group operations compared with the NAF_w method with $w=5$.

7. Conclusion

This paper showed the skew-Frobenius maps $\tilde{\phi}_q$ on the Jacobians of hyperelliptic curves of any genus. The map can be used for the expansion of an integer in $\mathbb{Z}[\tilde{\phi}_q]$ similar to the Frobenius maps. Then the maps can be more efficiently computable for hyperelliptic curves compared with elliptic curves. Furthermore, by using a rational point group of prime order, the operation cost of the definition field can be reduced. This paper also discussed the efficiency of the skew-Frobenius maps for scalar multiplications briefly.

An application of the proposed skew-Frobenius maps to scalar multiplications on the Jacobians of hyperelliptic curves needs a further study on efficient algorithms using the skew-Frobenius expansion and efficient implementations of the scalar multiplications.

Acknowledgement

The authors are grateful to the anonymous referee for her/his valuable comments.

References

[1] D.G. Cantor, "Computing in the Jacobian of hyperelliptic curve," Math. Comp., vol.48, no.177, pp.95–101, 1987.

[2] H. Cohen, G. Frey, R. Avanzi, C. Doche, T. Lange, K. Nguyen, and F. Vercauteren, Handbook of elliptic and hyperelliptic curve cryptography, Chapman & Hall/CRC, 2005.

[3] Y. Choie and J.W. Lee, "Speeding up the scalar multiplication in the Jacobians of hyperelliptic curves using Frobenius map," INDOCRYPT 2002, LNCS 2551, pp.285–295, 2002.

[4] S. Furihata, T. Kobayashi, and T. Saito, "Scalar multiplication on twist elliptic curves over $\mathbb{F}_{2^m}$," Proc. SCIS2003, pp.843–846, 2003.

[5] K. Furukawa, T. Kobayashi, and K. Aoki, "The cost of elliptic operations in OEF using a successive extension," Proc. SCIS2003, pp.929–934, 2003.

[6] M. Gonda, K. Matsuo, K. Aoki, J. Chao, and S. Tsujii, "Improvements of addition algorithm on genus 3 hyperelliptic curves and their implementation," IEICE Trans. Fundamentals, vol.E88-A, no.1, pp.89–96, Jan. 2005.

[7] T. Iijima, K. Matsuo, J. Chao, and S. Tsujii, "Cnstruction of Frobenius maps of twist elliptic curves and its application to elliptic scalar multiplication," Proc. SCIS2002, pp.699–702, 2002.

[8] N. Kanayama, K. Nagao, and S. Uchiyama, "Generating secure genus two hyperelliptic curves using Elkies' point counting algorithm," IEICE Trans. Fundamentals, vol.E86-A, no.4, pp.908–918, April 2003.

[9] N. Koblitz, "Hyperelliptic curve cryptosystems," J. Cryptol., vol.1, no.3, pp.139–150, 1989.

[10] S. Kozaki, K. Matsuo, and Y. Shimbara, "Skew-Frobenius maps on hyperelliptic curves," Proc. SCIS2007, 2007.

[11] T. Lange, "Koblitz curve cryptosystems," Finite Fields Appl., vol.11, no.2, pp.220–229, 2005.

[12] J. Nyukai, K. Matsuo, J. Chao, and S. Tsujii, "On the resultant computation in the addition Harley algorithms on hyperelliptic curves," IEICE Technical Report, ISEC2006-5, 2006.