

# Hyperelliptic pairings

Steven D. Galbraith<sup>1</sup>, Florian Hess<sup>2</sup>, and Frederik Vercauteren<sup>3\*</sup>

<sup>1</sup> Mathematics Department,  
Royal Holloway, University of London,  
Egham, Surrey, TW20 0EX, UK.  
`steven.galbraith@rhul.ac.uk`

<sup>2</sup> Technische Universität Berlin,  
Fakultät II, Institut für Mathematik Sekr. MA 8-1,  
Strasse des 17. Juni 136, D-10623 Berlin, Germany.  
`hess@math.tu-berlin.de`

<sup>3</sup> Department of Electrical Engineering,  
Katholieke Universiteit Leuven  
Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium  
`frederik.vercauteren@esat.kuleuven.be`

**Abstract.** We survey recent research on pairings on hyperelliptic curves and present a comparison of the performance characteristics of pairings on elliptic curves and hyperelliptic curves. Our analysis indicates that hyperelliptic curves are not more efficient than elliptic curves for general pairing applications.

## 1 Introduction

The original work on pairing-based cryptography [46, 47, 27, 6] used pairings on elliptic curves. It was then natural to suggest using higher genus curves for pairing applications [18]. The motivation given in [18] for considering higher genus curves was to have a wider choice of possible embedding degrees  $k$ . This motivation was supported by [42, 45] who showed that, for supersingular abelian varieties, one can always get larger security parameter in dimension 4 than using elliptic curves.

Koblitz [30] was the first to propose using divisor class groups of hyperelliptic curves for cryptosystems based on the discrete logarithm problem. Since that time there has been much research on comparing the speed of elliptic curves and hyperelliptic curves for cryptography. The current state of the art (see [7, 34] for a survey) suggests that in some situations genus 2 curves can be faster than elliptic curves. This gives further motivation for considering pairings on hyperelliptic curves.

Duursma and Lee [11] were the first to give fast algorithms for computing pairings on curves of genus  $\geq 2$ . Their loop shortening idea was generalised in [5, 26, 23]. Some other papers on hyperelliptic pairings are [10, 12, 35, 40].

---

\* Postdoctoral Fellow of the Research Foundation - Flanders (FWO)

It is therefore natural to explore whether pairings on hyperelliptic curves can be competitive/faster than pairings on elliptic curves, or whether there are any other advantages. This paper surveys the current state of knowledge on pairings on curves. We discuss possible advantages and disadvantages of using curves of genus  $g > 1$  and we present a number of open problems for future research.

The plan of the paper is as follows. We recall some background on hyperelliptic curves, divisor class groups, and representation of divisor classes. We discuss the supposed advantages of hyperelliptic vs. elliptic curves in standard cryptography (throughout the paper we use the phrase ‘non-pairing cryptography’ to denote the other applications of elliptic and hyperelliptic curves). We then recall the Tate-Lichtenbaum and ate pairings and present some implementation details for pairings on hyperelliptic curves, including a discussion of the critical computational task of evaluating a function at a divisor. We discuss the use of degenerate divisors for pairings, give some results on distortion maps for supersingular genus 2 curves and recall the Rubin-Silverberg point compression method. Finally, we give a thorough comparison of the performance characteristics of elliptic and hyperelliptic curves.

Our conclusion is that, for most applications, elliptic curves provide more efficient solutions than hyperelliptic curves. Nevertheless, there are many interesting open questions relating to pairings on hyperelliptic curves. In order to encourage research on this important topic we provide a list of problems for further study.

## 2 Background on curves

A good reference is [1]. An affine elliptic curve  $E$  over a finite field  $\mathbb{F}_q$  is given by an equation of the form

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

where  $a_1, a_3, a_2, a_4, a_6 \in \mathbb{F}_q$  are such that  $E$  is non-singular. An elliptic curve  $\mathcal{E}$  over  $\mathbb{F}_q$  is the associated projective curve of an affine elliptic curve  $E$ . It is a non-singular projective curve of genus one and has, in addition to the points of  $E$ , one extra point (called the point at infinity and denoted  $\infty$ ), which is  $\mathbb{F}_q$ -rational. The set of points  $\mathcal{E}(\mathbb{F}_q)$  forms a group, where  $\infty$  is the identity element and the group operation is given by the chord-and-tangent rule.

We define an affine hyperelliptic curve over  $\mathbb{F}_q$  to be a non-singular affine curve of the form

$$C : y^2 + H(x)y = F(x)$$

where  $H(x), F(x) \in \mathbb{F}_q[x]$ . We denote by  $g$  the genus of  $C$ , in which case we may assume that  $\deg(H(x)) \leq g + 1$  and  $\deg(F(x)) \leq 2g + 2$ . There is a single point at infinity on the associated projective curve  $\mathcal{C}_0$ , but it is now singular and there may be one  $\mathbb{F}_q$ -rational or two, not necessarily  $\mathbb{F}_q$ -rational points above this on the normalisation  $\mathcal{C}$  (i.e., desingularisation) of  $\mathcal{C}_0$ . We call  $\mathcal{C}$  a hyperelliptic curve over  $\mathbb{F}_q$  of genus  $g$ . For any extension field  $K$  of  $\mathbb{F}_q$  we denote by  $\mathcal{C}(K)$  the set of points on  $\mathcal{C}$  with coordinates in  $K$ .

## 2.1 The divisor class group

The points of  $\mathcal{C}$  no longer form a group, if  $g \geq 2$ . Instead, one works with the divisor class group of the curve.

The divisor group is defined as

$$\mathrm{Div}(\mathcal{C}) = \left\{ \sum_{P \in \mathcal{C}(\overline{\mathbb{F}}_q)} n_P(P) : n_P \in \mathbb{Z} \text{ and all but finitely many } n_P = 0 \right\},$$

where the sum is a formal sum over symbols  $(P)$ , and addition is carried out coefficientwise. For  $D \in \mathrm{Div}(\mathcal{C})$ , define  $\deg(D) = \sum_{P \in \mathcal{C}(\overline{\mathbb{F}}_q)} n_P \in \mathbb{Z}$  and  $v_P(D) = n_P$ , so we can write  $D = \sum_{P \in \mathcal{C}(\overline{\mathbb{F}}_q)} v_P(D)P$ . Let  $\mathrm{Div}^0(\mathcal{C}) = \{D \in \mathrm{Div}(\mathcal{C}) : \deg(D) = 0\}$  which is a subgroup of  $\mathrm{Div}(\mathcal{C})$ . The support of a divisor  $D$ , denoted  $\mathrm{supp}(D)$ , is the set  $\{P \in \mathcal{C}(\overline{\mathbb{F}}_q) : v_P(D) \neq 0\}$ . A divisor  $D$  is called effective if  $v_P(D) \geq 0$  for all  $P$ .

For any algebraic extension field  $K$  of  $\mathbb{F}_q$  a  $K$ -rational function on  $\mathcal{C}$  is a function  $f : \mathcal{C}(\overline{\mathbb{F}}_q) \rightarrow \overline{\mathbb{F}}_q \cup \{\infty\}$  that can be represented by a fraction  $g/h$  of homogeneous polynomials of the same degree defined over  $K$ . This means that for all  $P \in \mathcal{C}(\overline{\mathbb{F}}_q)$  we have either  $f(P) = g(P)/h(P)$  (evaluation of  $g, h$  at the coordinates of  $P$ ) or  $g(P) = h(P) = 0$ , and the latter happens for at most finitely many  $P$ . It can be shown that for  $P \in \mathcal{C}(\overline{\mathbb{F}}_q)$  with  $g(P) = h(P) = 0$  one can choose an alternative representation  $\tilde{g}/\tilde{h}$  of  $f$  such that  $\tilde{g}(P) \neq 0$  or  $\tilde{h}(P) \neq 0$ , hence  $f(P) = \tilde{g}(P)/\tilde{h}(P)$ .

The  $K$ -rational functions form a field  $K(\mathcal{C})$ , which is called the function field of  $\mathcal{C}$  over  $K$ . The function evaluation  $f(P)$  for  $f \in K(\mathcal{C})$  can be either zero, a non-zero value from  $\overline{\mathbb{F}}_q$ , or  $\infty$ . It is possible to associate a multiplicity  $v_P(f) \in \mathbb{Z}$  of zero (or pole), which satisfies the expected properties known from Laurent series. Equivalently, the function  $v_P$  is the valuation of the algebraic function field  $K(\mathcal{C})$  at the place  $P$ .

If  $f \in \overline{\mathbb{F}}_q(\mathcal{C})$  then one can define the divisor

$$\mathrm{div}(f) = \sum_{P \in \mathcal{C}(\overline{\mathbb{F}}_q)} v_P(f)(P).$$

It is a standard result that  $\deg(\mathrm{div}(f)) = 0$ , since  $\mathcal{C}$  is projective. The degree of  $f$  is defined as  $\deg(f) = \sum_{v_P(f) > 0} v_P(f) = -\sum_{v_P(f) < 0} v_P(f)$ .

The group of principal divisors is

$$\mathrm{Prin}(\mathcal{C}) = \{\mathrm{div}(f) : f \in \overline{\mathbb{F}}_q(\mathcal{C})\}$$

which is a subgroup of  $\mathrm{Div}^0(\mathcal{C})$ . The divisor class group is defined to be the quotient group

$$\mathrm{Pic}^0(\mathcal{C}) = \mathrm{Div}^0(\mathcal{C})/\mathrm{Prin}(\mathcal{C}).$$

Some authors write  $D_1 \sim D_2$  or  $D_1 \equiv D_2$  to represent equivalence of divisors in the quotient, in other words that there exists a function  $f$  such that  $D_1 =$

$D_2 + \text{div}(f)$ . The equivalence class containing a divisor  $D$  is called a divisor class and is denoted  $\overline{D}$ .

To obtain a finite group, we must consider divisor classes over  $\mathbb{F}_q$ , not  $\overline{\mathbb{F}}_q$ . A divisor  $D$  is said to be  $K$ -rational if  $D^\sigma = D$  for all  $\sigma \in \text{Gal}(\overline{\mathbb{F}}_q/K)$ . We define

$$\text{Div}_K^0(\mathcal{C}) = \{D \in \text{Div}^0(\mathcal{C}) : D^\sigma = D \ \forall \sigma \in \text{Gal}(\overline{\mathbb{F}}_q/K)\}.$$

We define  $\text{Prin}_K(\mathcal{C}) = \{\text{div}(f) : f \in K(\mathcal{C})\}$  and then

$$\text{Pic}_K^0(\mathcal{C}) = \text{Div}_K^0(\mathcal{C})/\text{Prin}_K(\mathcal{C})$$

and one can prove that this is isomorphic to the subgroup of  $\text{Pic}^0(\mathcal{C})$  which is invariant under  $\text{Gal}(\overline{\mathbb{F}}_q/K)$ . For  $r \in \mathbb{N}$  we define

$$\text{Pic}_K^0(\mathcal{C})[r] = \{D \in \text{Pic}_K^0(\mathcal{C}) : rD = 0\}.$$

The Riemann hypothesis for function fields (proved by Weil) implies that  $\#\text{Pic}_{\mathbb{F}_q}^0(\mathcal{C}) \approx q^g$  for  $q$  large and  $g$  small. This is the primary motivation for considering hyperelliptic curves for cryptography: for a  $k$ -bit group size one can work over a finite field of about  $k/g$  bits.

The Jacobian of a curve  $\mathcal{C}$  is the abelian variety  $\text{Jac}(\mathcal{C})$  that contains  $\mathcal{C}$  and has the property that, for every extension field  $K$  of  $\mathbb{F}_q$ , the groups  $\text{Jac}(\mathcal{C})(K)$  and  $\text{Pic}_K^0(\mathcal{C})$  are isomorphic.

## 2.2 Mumford representation

In this section we assume  $\mathcal{C}$  is a hyperelliptic curve of genus  $g$  over  $K$  with a single  $K$ -rational point  $\infty$  at infinity. To be able to explicitly compute with elements of  $\text{Pic}_K^0(\mathcal{C})$  we have to choose a compact representation of the elements. To this end we introduce the following notion.

**Definition 1.** A divisor  $D \in \text{Div}_K^0(\mathcal{C})$  on a hyperelliptic curve  $\mathcal{C}$  of genus  $g$  is called semi-reduced if it can be written as  $D = E - d(\infty)$  with  $E$  effective and for  $P = (x, y)$  with  $2y + H(x) = 0$  one has  $v_P(E) \in \{0, 1\}$ , and for  $P = (x, y)$  and  $P' = (x, -y - H(x))$  with  $P \neq P'$  (equivalently  $2y + H(x) \neq 0$ ) one has  $v_P(E)v_{P'}(E) = 0$ . If moreover  $\deg(E) \leq g$  then  $D$  is called reduced.

Every divisor class in  $\text{Pic}_K^0(\mathcal{C})$  contains exactly one reduced divisor. A reduced divisor  $D = E - d(\infty)$  is represented in Mumford representation as a pair  $[u(x), v(x)]$  of polynomials in  $K[x]$  such that:  $u(x)$  is monic,  $u(x)$  divides  $F(x) - H(x)v(x) - v(x)^2$ ,  $\deg(v(x)) < \deg(u(x)) \leq g$ . Subject to these conditions, the relation between  $E = \sum_{i=1}^d (x_i, y_i)$  and  $[u(x), v(x)]$  is as follows:  $u(x) = \prod_{i=1}^d (x - x_i)$  and  $v(x_i) = y_i$ . This yields a 1-1 correspondence between reduced divisors and their Mumford representation.

---

**Algorithm 1** Cantor Addition

---

INPUT: Divisors  $D_1 = [u_1, v_1]$  and  $D_2 = [u_2, v_2]$ OUTPUT: A reduced divisor  $D$  representing the sum  $D_1 + D_2$  in  $\text{Pic}_K^0(\mathcal{C})$ .

```
1:  $d_1 \leftarrow \gcd(u_1, u_2) = e_1 u_1 + e_2 u_2$ 
2:  $d \leftarrow \gcd(d_1, v_1 + v_2 + H) = c_1 d_1 + c_2(v_1 + v_2 + H)$ 
3:  $s_1 \leftarrow c_1 e_1, s_2 \leftarrow c_1 e_2, s_3 \leftarrow c_2$ 
4:  $u \leftarrow (u_1 u_2)/d^2, v \leftarrow (s_1 u_1 v_2 + s_2 u_2 v_1 + s_3(v_1 v_2 + F))/d \bmod u$ 
5: while  $\deg(u) > g$  do
6:    $u' \leftarrow \text{Monic}((F - vH - v^2)/u), v' \leftarrow (-H - v) \bmod u'$ 
7:    $u \leftarrow u', v \leftarrow v'$ 
8: end while
9: return  $D = [u', v']$ .
```

---

### 2.3 Cantor's algorithm

We continue to assume that  $\mathcal{C}$  is a hyperelliptic curve over  $K$  with a single  $K$ -rational point at infinity. An algorithm for adding general divisor classes in this case was developed by Cantor [9] for the case that  $H(x) = 0$  and the characteristic of  $K$  is not 2. The general case was worked out by Koblitz in [30].

Cantor's algorithm is given in Algorithm 1, but as presented here, the addition algorithm is not very efficient, since it requires an extended Euclidean gcd computation in Steps 1 and 2. However if one fixes the genus  $g$ , one can work out specific algorithms dedicated to the various possible values of  $\deg(u_1)$  and  $\deg(u_2)$  [25, 32, 33]. This way one can formulate algorithms that are much more efficient, and that avoid high-level operations like Euclidean algorithms.

The relation with divisor equivalence is as follows: in Step 4 we obtain a semi-reduced divisor  $D$  represented by  $[u(x), v(x)]$  that satisfies

$$D = D_1 + D_2 - \text{div}(d(x)). \quad (1)$$

The divisor  $D$  is then further reduced in the loop in Step 5 to a divisor  $D'$  represented by  $[u'(x), v'(x)]$  which satisfies

$$D' = D - \text{div}((y - v(x))/u'(x)). \quad (2)$$

## 3 Elliptic versus hyperelliptic curves

The primary motivation for considering curves of genus  $g > 1$  for cryptography is the fact that the group size grows as  $q^g$ . In other words, with genus 2 one can get a given group size by working over a field  $\mathbb{F}_q$  where  $q$  has half the bit length needed if working with elliptic curves.

There has been much discussion about whether or not genus 2 curves can be faster for non-pairing cryptography than elliptic curves. To get comparable timings it is crucial to replace Cantor's algorithm with some optimised formulae [25, 32, 33]. Nevertheless, it seems that fully general implementations of genus 2 curves are slower than general implementations of elliptic curves.

However, there are several special tricks available for genus 2 curves, which have no counterpart for elliptic curves. We will briefly mention two such tricks here.

The first is to use curves (in characteristic 2) of the special form

$$y^2 + xy = x^5 + F_3x^3 + F_2x^2 + F_0.$$

For these curves, the optimised group law formulae require one inversion, 6 squarings and 5 multiplications [33, 34], which is very competitive compared with elliptic curves. For pairings using supersingular curves in characteristic 2 one has curves which are even more special, and thus faster, than this. One can avoid inversions if some form of projective coordinates is used.

A second trick is to utilise special divisors. Recall that a general reduced divisor  $D$  on a genus 2 curve has support consisting of two affine points (i.e.,  $D = (P_1) + (P_2) - 2(\infty)$ ). Following [28, 29] one can exploit the benefits of using *degenerate divisors* of the form  $D = (P) - (\infty)$ . (Note that the definition of degenerate divisors in [28, 29] is that they have less than  $g$  points in their support, whereas our definition is more restrictive when  $g > 2$  in that we insist on having exactly one affine point in the support.) The addition operations are faster when adding a general divisor to a degenerate divisor, than when adding two general divisors.

To summarise, hyperelliptic curves can be competitive with elliptic curves (sometimes, even faster) due to the smaller field size, as long as one exploits optimised addition formulae for special curves and one uses special divisors.

## 4 A world of pairings

### 4.1 Weil and Tate-Lichtenbaum pairings

Let  $\mathcal{C}$  be a non-singular projective curve of genus  $g$  over  $\mathbb{F}_q$ . Let  $r$  be coprime to  $q$ . It is typical for cryptographic applications to take  $r$  to be a (large) prime divisor of  $\#\text{Pic}_{\mathbb{F}_q}^0(\mathcal{C})$ . It is often the case that  $r \approx q^g$ , but in some situations it is necessary to take  $r$  smaller. The embedding degree is defined to be the smallest positive integer  $k$  such that  $r \mid (q^k - 1)$ . Note that the embedding degree is a function of  $q$  and  $r$ . The subgroup of  $r$ -th roots of unity of  $\mathbb{F}_{q^k}^\times$  is denoted by  $\mu_r = \{z \in \mathbb{F}_{q^k}^\times : z^r = 1\}$ .

The Weil pairing [52, 39] is defined to be a non-degenerate bilinear map

$$\text{Pic}_{\mathbb{F}_q}^0(\mathcal{C})[r] \times \text{Pic}_{\mathbb{F}_q}^0(\mathcal{C})[r] \longrightarrow \mu_r$$

which is denoted  $e_r(\overline{D}_1, \overline{D}_2)$ .

The Tate-Lichtenbaum pairing [49, 36, 16] is defined to be a non-degenerate bilinear map

$$\text{Pic}_{\mathbb{F}_{q^k}}^0(\mathcal{C})[r] \times \text{Pic}_{\mathbb{F}_{q^k}}^0(\mathcal{C})/r\text{Pic}_{\mathbb{F}_{q^k}}^0(\mathcal{C}) \longrightarrow \mathbb{F}_{q^k}^\times/(\mathbb{F}_{q^k}^\times)^r$$

which is denoted  $\langle \overline{D}_1, \overline{D}_2 \rangle_r$ .

The domain and range of the Weil pairing are better suited for cryptographic applications, since the pairing arguments and values are given by points and finite field elements instead of equivalence classes. For many cryptographic applications it is necessary to work with unique representatives (e.g., Alice and Bob may perform different calculations but must obtain the same element). To achieve this for the Tate-Lichtenbaum pairing, two simplifications are usually made. First, if we assume that  $\text{Pic}_{\mathbb{F}_{q^k}}^0(\mathcal{C})$  contains no elements of order  $r^2$  then we may identify  $\text{Pic}_{\mathbb{F}_{q^k}}^0(\mathcal{C})[r]$  with  $\text{Pic}_{\mathbb{F}_{q^k}}^0(\mathcal{C})/r\text{Pic}_{\mathbb{F}_{q^k}}^0(\mathcal{C})$ , via the map  $\overline{D}_2 \mapsto \overline{D}_2 + r\text{Pic}_{\mathbb{F}_{q^k}}^0(\mathcal{C})$ . Second, we can map into the subgroup  $\mu_r$  by raising to the power  $(q^k-1)/r$ . This is called the final exponentiation. Hence, we consider the reduced (or modified) Tate-Lichtenbaum pairing

$$t(\overline{D}_1, \overline{D}_2) = \langle \overline{D}_1, \overline{D}_2 \rangle_r^{(q^k-1)/r}.$$

The mathematical definition of the Tate-Lichtenbaum pairing is as follows. The argument on the left hand side of the Tate-Lichtenbaum pairing is represented by an  $\mathbb{F}_{q^k}$ -rational divisor  $D_1$  of degree zero. Since  $\overline{D}_1$  is a divisor class of order  $r$ , there is a function  $f_{r,D_1}$  with divisor

$$\text{div}(f_{r,D_1}) = rD_1.$$

The argument of the right hand side of the Tate-Lichtenbaum pairing can be represented by an  $\mathbb{F}_{q^k}$ -rational divisor  $D_2$  of degree zero such that the supports of  $D_1$  and  $D_2$  are disjoint. Then the Tate-Lichtenbaum pairing is defined to be

$$\langle \overline{D}_1, \overline{D}_2 + r\text{Pic}_{\mathbb{F}_{q^k}}^0(\mathcal{C}) \rangle_r = f_{r,D_1}(D_2) = \prod_P f_{r,D_1}(P)^{v_P(D_2)}.$$

The Weil pairing usually offers inferior efficiency and flexibility in comparison with the reduced Tate-Lichtenbaum pairing (see for example [24] or [1, Section 16.1.5]). In the remainder of the paper we will therefore consider the reduced Tate pairing only. More information about the Weil pairing and its efficient computation can be found in [38].

Finally, we note that  $f_{r,D}$  with  $\text{div}(f_{r,D}) = rD$  is only defined up to scalar multiples from  $\overline{\mathbb{F}}_q^\times$ . It is possible to find  $f_{r,D}$  which is defined over the field of definition of  $D$  and we assume this in the following. We will need to impose some additional normalisation conditions on  $f_{r,D}$  later.

## 4.2 Ate pairings

For cryptographic purposes one applies one further simplification to the reduced Tate-Lichtenbaum pairing by restricting the pairing to certain cyclic subgroups  $G_1$  and  $G_2$  of  $\text{Pic}_{\mathbb{F}_{q^k}}^0(\mathcal{C})[r]$  that are Frobenius eigenspaces. Write  $\pi$  for the  $q$ -power Frobenius map on  $\mathcal{C}$  and the Frobenius endomorphism on  $\text{Pic}_{\mathbb{F}_{q^k}}^0(\mathcal{C})$ . Then we define

$$G_1 = \text{Pic}_{\mathbb{F}_q}^0(\mathcal{C})[r],$$

for which the eigenvalue of  $\pi$  is 1. We also define

$$G_2 = \text{Pic}_{\mathbb{F}_{q^k}}^0(\mathcal{C})[r] \cap \ker(\pi - q).$$

The Weil and Tate-Lichtenbaum pairings are the two known, essentially different pairings available for curves. The elliptic and hyperelliptic ate pairings can be regarded as special, low degree variants of the Tate-Lichtenbaum pairing. Note that while the Weil and Tate-Lichtenbaum pairings are named after their inventors and are already quite classical, the ate pairing is much more recent and carries an artificial name.

**Ate pairings on elliptic curves** Let  $\mathcal{E}$  be an ordinary elliptic curve over  $\mathbb{F}_q$ . Let  $t$  be the trace of the  $q$ -power Frobenius endomorphism  $\pi$  of  $\mathcal{E}$ , such that  $\#\mathcal{E}(\mathbb{F}_q) = q - t + 1$ . We assume that  $r \geq 5$  is a sufficiently large prime factor of  $\#\mathcal{E}(\mathbb{F}_q)$  and that  $k$  is minimal such that  $r \mid (q^k - 1)$ .

If  $P \in \mathcal{E}(\overline{\mathbb{F}}_q)$  has order  $r$ , then  $(P) - (\infty)$  is a divisor of degree zero representing a divisor class of order  $r$ . We define  $f_{r,P} = f_{r,(P)-(\infty)}$ . Recall our assumption that the field of definition of  $f_{r,P}$  be that of  $P$ . In addition to this, we need to normalise  $f_{r,P}$  as follows. Let  $z \in \mathbb{F}_q(\mathcal{E})$  be a local uniformizer at  $\infty$ , that is  $z$  satisfies  $v_\infty(z) = 1$ . Then we define  $\text{lc}_\infty(f_{r,P}) = (z^r f_{r,P})(\infty)$  and  $f_{r,P}^{\text{norm}} = f_{r,P} / \text{lc}_\infty(f_{r,P})$ . The function  $f_{r,P}^{\text{norm}}$  is defined over the field of definition  $K$  of  $P$  and is uniquely determined by  $r$  and  $P$  up to non-zero  $r$ th-power multiples from  $K$ .

The following theorem is from [26], using a slightly more general formulation given in [37].

**Theorem 1.** *Let  $S$  be an integer with  $S \equiv q \pmod{r}$ . Define  $N = \gcd(S^k - 1, q^k - 1)$  and  $L = (S^k - 1)/N$ . Let  $c_S = \sum_{i=0}^{k-1} S^{k-1-i} q^i \pmod{N}$ . Then*

$$a_S : G_2 \times G_1 \rightarrow \mu_r, \quad (Q, P) \mapsto f_{S,Q}^{\text{norm}}(P)^{c_S(q^k-1)/N}$$

*defines a bilinear pairing, called the elliptic ate pairing. If  $k \mid \#\text{Aut}(\mathcal{E})$  then*

$$a_S^{\text{twist}} : G_1 \times G_2 \rightarrow \mu_r, \quad (P, Q) \mapsto f_{S,P}(Q)^{c_S(q^k-1)/N}$$

*also defines a bilinear pairing, called the twisted ate pairing. Both pairings  $a_S$  and  $a_S^{\text{twist}}$  are non-degenerate if and only if  $r \nmid L$ .*

*The relation with the reduced Tate-Lichtenbaum pairing is*

$$a_S(Q, P) = t(Q, P)^L \quad \text{and} \quad a_S^{\text{twist}}(P, Q) = t(P, Q)^L.$$

We remark that the condition  $k \mid \#\text{Aut}(\mathcal{E})$  holds true if and only if  $\mathcal{E}$  admits a twist of degree  $k$ . We say that  $\mathcal{E}$  admits a twist of degree  $d$  if there is an elliptic curve  $\mathcal{E}'$  defined over  $\mathbb{F}_q$  and an isomorphism  $\psi : \mathcal{E}' \rightarrow \mathcal{E}$  defined over  $\mathbb{F}_{q^d}$ , and  $d$  is minimal with this property. If  $k \mid \#\text{Aut}(\mathcal{E})$  does not hold one may still apply the theorem for a divisor  $e$  of  $k$  using a base extension of degree  $k/e$ .

One can take  $S = q$  in Theorem 1, but the usual choice is  $S = t - 1$ , which has half the bit length of  $\#\mathcal{E}(\mathbb{F}_q)$  and thus yields half the loop length of the standard reduced Tate-Lichtenbaum pairing, if  $r \approx q$ . In certain cases it may be possible to choose  $S$  strictly smaller than  $t - 1$ , which yields an even more efficient computation [37].

The Duursma-Lee pairing [11] and the  $\eta_T$ -pairing from [5] can be regarded as a special form of the twisted ate pairing on supersingular elliptic curves.

**Ate pairings on hyperelliptic curves** For hyperelliptic curves the situation is somewhat different. Indeed, if  $\mathcal{C}$  is a hyperelliptic curve then  $r \mid \#\text{Pic}_{\mathbb{F}_q}^0(\mathcal{C}) = q^g + a_1(q^{g-1} + 1) + a_2(q^{g-2} + 1) + \dots + a_g$ . If  $r \approx \#\text{Pic}_{\mathbb{F}_q}^0(\mathcal{C})$  then the bit length of  $q$  is already  $g$  times shorter than the bit length of  $r$ , so we may try to mimic Theorem 1 with  $S = q$  (this idea, in a special case, first appears in [11]).

In order to formulate the main results from [23], we fix some notation. Let  $\mathcal{C}$  be a hyperelliptic curve with a single point  $\infty$  at infinity. For any divisor class  $\overline{D}$  we denote by  $\rho(\overline{D})$  the unique reduced divisor in  $\overline{D}$  and by  $\epsilon(\overline{D})$  the effective part of  $\rho(\overline{D})$  so that we have  $\rho(\overline{D}) = \epsilon(\overline{D}) - d(\infty)$ . We apply the same normalisation to the function  $f_{r,D}$  as above, namely  $f_{r,D}^{\text{norm}} = f_{r,D}/(\text{lc}_{\infty}(f_{r,D}))$  for  $\text{lc}_{\infty}(f_{r,P}) = (z^r f_{r,P})(\infty)$  and  $z \in \mathbb{F}_q(\mathcal{C})$  a local uniformizer at  $\infty$  over  $\mathbb{F}_q$ . A curve is called superspecial if its Jacobian is isomorphic to  $E^g$  with  $E$  a supersingular elliptic curve. The Jacobian of superspecial curves is hence also supersingular, and in particular has  $p$ -rank zero.

**Theorem 2.** ([23]) *With the above notation and assumptions,*

$$a : G_2 \times G_1 \rightarrow \mu_r : (\overline{D}_2, \overline{D}_1) \mapsto f_{q,\rho(\overline{D}_2)}^{\text{norm}}(\epsilon(\overline{D}_1))$$

*defines a non-degenerate, bilinear pairing called the hyperelliptic ate pairing. If  $\mathcal{C}$  is superspecial and  $d = \gcd(k, q^k - 1)$  then*

$$\hat{a} : G_1 \times G_2 \rightarrow \mu_r : (\overline{D}_1, \overline{D}_2) \mapsto f_{q,\rho(\overline{D}_1)}^{\text{norm}}(\epsilon(\overline{D}_2))^d$$

*defines a non-degenerate, bilinear pairing.*

*If in any of the two pairings we have  $\text{supp}(\epsilon(\overline{D}_i)) \cap \text{supp}(\rho(\overline{D}_j)) \neq \emptyset$ , then  $\epsilon(\overline{D}_i)$  needs to be replaced by any  $D \in \overline{D}_i$  with  $\text{supp}(D) \cap \text{supp}(\rho(\overline{D}_j)) = \emptyset$ .*

*The relation with the reduced Tate-Lichtenbaum pairing is*

$$t(\overline{D}_2, \overline{D}_1) = a(\overline{D}_2, \overline{D}_1)^{kq^{k-1}} \quad \text{and} \quad t(\overline{D}_1, \overline{D}_2) = \hat{a}(\overline{D}_1, \overline{D}_2)^{(k/d)q^{k-1}}.$$

One feature of the hyperelliptic ate pairing is that the final exponentiation is very simple.

## 5 Pairing friendly curves

A curve  $\mathcal{C}$  over  $\mathbb{F}_q$  of genus  $g$  is called pairing friendly if there is a large prime  $r \mid \#\text{Pic}_{\mathbb{F}_q}^0(\mathcal{C})$  with relatively small embedding degree  $k$  (say,  $2 \leq k \leq 30g$ ) and such that  $\mu_r$  does not lie in a proper subfield of  $\mathbb{F}_{q^k}$ . Supersingular curves are pairing friendly. In this section we list some curves which are particularly suitable for efficient pairing implementation.

## 5.1 Elliptic curves

The most useful cases are ordinary curves for which the value  $S$  in the elliptic ate pairing is small, and supersingular curves. An example in the supersingular case is the curve  $E : y^2 + y = x^3 + x + b$  over  $\mathbb{F}_{2^m}$  where  $b = 0, 1$  and  $m$  is odd. The group order is  $2^m + 1 \pm 2^{(m+1)/2}$  and the embedding degree is  $k = 4$ . See [18, 5] for more details.

Elliptic curves suitable for the elliptic ate pairing can be constructed using the method of Brezing and Weng [8]. A family which is especially adapted to the ate pairing is given by the following parameterisation (which for  $n = 1$  was given already in [3]): let  $r(x) = \Phi_{12n}(x)$  for any positive integer  $n$  coprime to 3,  $t(x) = x + 1$  and  $s(x) = (2x^{2n} - 1)$ . Following the method of [8] we define  $p(x) = (t(x)^2 - s(x)^2(t(x) - 2)^2/D)/4$  with  $D = -3$ . If  $x$  is such that  $p(x)$  and  $r(x)$  are prime, then one can easily construct an elliptic curve  $y^2 = x^3 + b$  with embedding degree  $12n$ . For  $n = 1$  then  $\deg(r(x)) = 4$  and  $\deg(p(x)) = 6$ , but for larger values of  $n$  one can get  $\deg(r(x)) \approx \deg(p(x))$ . Note that the trace is as small as possible compared to  $r$  and that the elliptic curve admits sextic twists; both these features are desirable for fast implementations of the ate pairings.

## 5.2 Hyperelliptic curves

An example of a supersingular genus 2 curve is  $C_d : y^2 + y = x^5 + x^3 + d$  with  $d = 0$  or  $1$  over  $\mathbb{F}_{2^m}$ , where  $m$  is coprime to 6. This curve has embedding degree 12 (see [18, 5] for more details).

An example of a family of superspecial hyperelliptic curves was studied by Duursma and Lee [11]. They considered the curves  $C : y^2 = x^p - x + d$  over  $\mathbb{F}_{p^m}$  where  $d = \pm 1$ . The genus of  $C$  is  $(p - 1)/2$ . When  $p \equiv 3 \pmod{4}$  the embedding degree is  $k = 2p$ . If  $p \equiv 1 \pmod{4}$  then the embedding degree is  $k = p$ . It is worth noting that the fast pairing algorithms of [11, 5] required the condition  $p \equiv 3 \pmod{4}$ , but the ate method works for all cases.

In characteristic  $p \geq 5$ , the best one can do with supersingular genus 2 curves is  $k = 6$ . In [20] it is shown how to obtain suitable curves for any  $p \equiv 2 \pmod{3}$  by taking twists of the supersingular curve  $y^2 = x^6 + 1$ .

It is natural to try to use non-supersingular curves of genus  $g \geq 2$  for pairing-based cryptography. However, it seems to be hard to generate suitable curves in this case. The paper [19] gives some first steps towards solving this problem, by presenting some quadratic polynomial families of abelian surfaces with given embedding degree. However, [19] proves that for some embedding degrees (e.g.,  $k = 8$  and  $k = 12$ ) there are no such quadratic families. These results suggest that there is less structure in the genus 2 case (or, at least, that the structure is more complicated in the genus 2 case) than in the elliptic case. For the polynomial families found in [19] the authors were unable to generate any curves using the CM method. Indeed, the CM method for curves of genus  $\geq 2$  is much less well developed than the CM method in the elliptic case.

The first examples of non-supersingular pairing-friendly genus 2 curves are due to Freeman [14]. The parameters for these curves are not very attractive for

fast pairing implementation (precisely, the size of  $r$  is too small compared with the size of  $q$ ). More research is needed on methods to generate such curves with parameters suitable for cryptography.

Note that many of the computational assumptions in pairing based cryptography can be solved in subexponential time, hence it may not be necessary to restrict to very small genus  $g$ , as is usually done for non-pairing cryptography. Nevertheless, in our comparison with elliptic curves we will assume that  $g \leq 5$ .

## 6 Miller's algorithm

This section reviews the generalisation of Miller's algorithm for computing Tate-Lichtenbaum or ate pairings efficiently on hyperelliptic curves  $y^2 + H(x)y = F(x)$  over  $\mathbb{F}_q$  with a single point at infinity. We make the following assumptions:

- We are pairing two reduced divisors  $D_1$  and  $D_2$  with Frobenius eigenvalues 1 and  $q$  (for the Tate-Lichtenbaum pairing,  $D_1$  has eigenvalue 1 and  $D_2$  has eigenvalue  $q$ , for the ate pairing it is the other way around).
- The pairing is computed as  $f_{S, D_1}(\epsilon(\overline{D}_2))^d$  for some integers  $S$  and  $d$ .

### 6.1 Hyperelliptic Miller

Miller's algorithm computes functions  $f_{n, D}$  with divisor  $\text{div}(f_{n, D}) = nD - D_n$ , where  $D_n = \rho(n\overline{D})$ , i.e. the unique reduced divisor equivalent to  $nD$ . It basically consists of a double and add algorithm exploiting the fact that one can define

$$f_{k+l, D} = f_{k, D} \cdot f_{l, D} \cdot h_{D_k, D_l},$$

with  $\text{div}(h_{D_k, D_l}) = D_k + D_l - \rho(\overline{D}_k + \overline{D}_l)$ . These functions  $h_{D_k, D_l}$  are obtained easily from Cantor's algorithm by equations (1) and (2).

However, we are not really interested in the functions  $f_{n, D_1}$  themselves, but only in the evaluation  $f_{n, D_1}(E)$  for some effective divisor  $E$ , so we need a method to evaluate  $h_{D_k, D_l}$  at  $E$ . Since this evaluation step is the crucial part of Miller's algorithm, we describe two different methods in detail (namely using a norm computation and using resultants). The papers [23, 35] use resultants for pairing computation.

Since any function can be written as a fraction of two polynomials, we can limit ourselves to evaluating a polynomial  $h(x, y) \in \mathbb{F}_q[x, y]$  at  $E$  for some  $\mathbb{F}_q$ -rational divisor  $D = E - d(\infty)$ , with Mumford representation  $[u_E(x), v_E(x)]$ .

The first method is a simple optimisation of the definition of function evaluation at a divisor. Let  $E = \sum_{i=1}^d (x_i, y_i)$ , then we can compute the support of  $E$  by factoring  $u_E(x)$  as  $\prod_{i=1}^d (x - x_i)$  and setting  $y_i = v_E(x_i)$ . In general each  $(x_i, y_i)$  will be defined over some extension field  $\mathbb{F}_{q^{e_i}}$ , where  $e_i \leq g$ . Of course, we could then compute  $h(E)$  as  $\prod_{i=1}^d h(x_i, y_i)$ , but in general this is not the best method, since we are not exploiting the fact that the result has to be in  $\mathbb{F}_q$ . Instead, one could partition the support into distinct Galois orbits

$$\left\{ (x_i, y_i), (x_i^q, y_i^q), \dots, (x_i^{q^{e_i-1}}, y_i^{q^{e_i-1}}) \right\}$$

and compute the product of the evaluations at these points simply by computing  $N_{\mathbb{F}_{q^{e_i}}/\mathbb{F}_q}(h(x_i, y_i))$ . The above method is in general suboptimal due to the overhead of factoring the polynomial  $u_E(x)$ . Therefore, we advise to restrict its use to the case of degenerate divisors.

The second method is in general faster than the first, since it does not involve any polynomial factorisations (nor any explicit arithmetic in extension fields the way we have described it). It is based on the following basic observation: the univariate polynomial  $\tilde{h}(x) = h(x, v_E(x))$  satisfies  $\tilde{h}(x_i) = h(x_i, y_i)$ , so we have reduced the problem to computing  $\prod_{i=1}^d \tilde{h}(x_i)$  where the  $x_i$  are the zeros of the monic polynomial  $u_E(x)$ . But this corresponds to the very definition of the resultant of the two polynomials  $u_E(x)$  and  $\tilde{h}(x)$ , so we conclude

$$h(E) = \text{Res}(u_E(x), h(x, v_E(x))).$$

Note that the above resultant also equals  $\text{Res}(u_E(x), \tilde{h}(x) \bmod u_E(x))$ , so it suffices to work with polynomials of degree smaller than  $g$ . In more mathematical terms, we still compute  $h(E)$  by using a norm, namely  $h(E) = N_{A/\mathbb{F}_q}(h(x, y))$  with  $A$  the finite  $\mathbb{F}_q$ -algebra  $\mathbb{F}_q[x, y]/(u_E(x), y - v_E(x))$ .

Algorithm 2 below executes one step in Miller's algorithm and is a simple adaptation of Cantor's algorithm. It computes the evaluation of  $h_{D_1, D_2}^{\text{norm}}(E)$  represented as follows: assume that  $h_{D_1, D_2} = h_1(x, y)/h_2(x, y)$ , then the algorithm returns  $\tilde{h}_1(x) = h_1(x, v_E(x)) \bmod u_E(x)$  and  $\tilde{h}_2(x) = h_2(x, v_E(x)) \bmod u_E(x)$  and the constant  $h_3 = \text{lc}_\infty(h_{D_1, D_2})$ , so we conclude that

$$h_{D_1, D_2}^{\text{norm}}(E) = \text{Res}(u_E(x), \tilde{h}_1(x)) / (h_3^{\deg(u_E)} \cdot \text{Res}(u_E(x), \tilde{h}_2(x))).$$

We assume in Algorithm 2 that all the intermediate functions are defined on  $\epsilon(E)$  (this is a reasonable assumption for the cryptographic applications).

These partial evaluations are then combined in Algorithm 3 below using a double and add strategy. Note that the resultant computation is postponed to the very end of the algorithm. The alternate strategy of computing the resultant each time instead of computing modulo  $u_E(x)$  is faster only in the genus 2 case.

## 6.2 Improvements to Miller's algorithm

There are a number of standard implementation techniques to speed up pairing computation (see [2, 4, 11, 17, 22]). We always assume that the embedding degree satisfies  $k > 1$ . The improvements include the following.

- Using suitable representations for  $\mathbb{F}_{q^k}$ , such as pairing friendly fields [31], i.e.  $\mathbb{F}_q$  is a prime field with  $q \equiv 1 \pmod{12}$  and  $k$  of the form  $2^i 3^j$ . Using a combination of Karatsuba and Toom-Cook, multiplication in such fields takes  $3^i 5^j$  multiplications in the base field  $\mathbb{F}_q$ .
- Changing the base in Miller's algorithm. For example, base 3 is used in [2, 17] and base 8 is used in [5] in genus 2.

---

**Algorithm 2** Miller Step

---

INPUT:  $D_1 = [u_1, v_1]$ ,  $D_2 = [u_2, v_2]$ ,  $E = [u_E, v_E]$ .OUTPUT: Reduced divisor  $\rho(\overline{D_1} + \overline{D_2})$  and evaluation  $h_{D_1, D_2}^{\text{norm}}(E)$ , represented by  $[\tilde{h}_1(x), \tilde{h}_2(x), h_3]$ .

```
1:  $d_1 \leftarrow \gcd(u_1, u_2) = e_1 u_1 + e_2 u_2$ 
2:  $d \leftarrow \gcd(d_1, v_1 + v_2 + H) = c_1 d_1 + c_2(v_1 + v_2 + H)$ 
3:  $\tilde{h}_1 \leftarrow d \bmod u_E, \tilde{h}_2 \leftarrow 1, h_3 \leftarrow 1$ 
4:  $s_1 \leftarrow c_1 e_1, s_2 \leftarrow c_1 e_2, s_3 \leftarrow c_2$ 
5:  $u \leftarrow (u_1 u_2)/d^2, v \leftarrow (s_1 u_1 v_2 + s_2 u_2 v_1 + s_3(v_1 v_2 + F))/d \bmod u$ 
6: while  $\deg(u) > g$  do
7:    $u' \leftarrow \text{Monic}((F - vH - v^2)/u), v' \leftarrow (-H - v) \bmod u'$ 
8:    $\tilde{h}_1 \leftarrow \tilde{h}_1 \cdot (v_E - v) \bmod u_E$ 
9:    $\tilde{h}_2 \leftarrow \tilde{h}_2 \cdot u' \bmod u_E$ 
10:  if  $\deg(v) > g$  then
11:     $h_3 \leftarrow -\text{lc}(v) \cdot h_3$  ▷ lc = leading coefficient
12:  end if
13:   $u \leftarrow u', v \leftarrow v'$ 
14: end while
15: return  $[u, v], [\tilde{h}_1, \tilde{h}_2, h_3]$ 
```

---

---

**Algorithm 3** Miller's algorithm (base 2)

---

INPUT:  $S = \sum_{i=0}^B S_i 2^i$ ,  $d$ ,  $D_1 = [u_1, v_1]$ ,  $D_2 = [u_2, v_2]$ .OUTPUT: Pairing value  $f_{S, D_1}(\epsilon(\overline{D_2}))^d$ 

```
1:  $D \leftarrow [u_1, v_1]$ 
2:  $f \leftarrow 1, f_1 \leftarrow 1, f_2 \leftarrow 1, f_3 \leftarrow 1$ 
3: for  $i \leftarrow B - 1$  downto 0 do
4:    $f_1 \leftarrow f_1^2 \bmod u_2, f_2 \leftarrow f_2^2 \bmod u_2, f_3 \leftarrow f_3^2$ 
5:    $D, [h_1, h_2, h_3] \leftarrow \text{Miller Step}(D, D, D_2)$ 
6:    $f_1 \leftarrow f_1 \cdot h_1 \bmod u_2, f_2 \leftarrow f_2 \cdot h_2 \bmod u_2, f_3 \leftarrow f_3 \cdot h_3$ 
7:   if  $S_i = 1$  then
8:      $D, [h_1, h_2, h_3] \leftarrow \text{Miller Step}(D, D_1, D_2)$ 
9:      $f_1 \leftarrow f_1 \cdot h_1 \bmod u_2, f_2 \leftarrow f_2 \cdot h_2 \bmod u_2, f_3 \leftarrow f_3 \cdot h_3$ 
10:  end if
11: end for
12:  $f \leftarrow \text{Res}(u_2, f_1)/(f_3^{\deg(u_2)} \cdot \text{Res}(u_2, f_2))$ 
13: return  $f^d$ 
```

---

- Working with divisors of the form  $\epsilon(\overline{D}_2)$  and computing leading coefficients instead of evaluating functions on the reduced divisor  $D_2$  itself. Since the support of  $D_2$  has  $\infty$  in common with the support of  $D_1$ , the alternative is to “shift”  $D_2$  and this leads to extra function evaluations.
- Denominator elimination [11, 5]. If a final exponentiation is performed then one can omit all terms lying in a proper subfield  $K$  such that  $\mathbb{F}_q \subseteq K \subset \mathbb{F}_{q^k}$ . In many cases (e.g. if one has even embedding degree, and  $D_1$  and  $D_2$  lie in the 1-eigenspace and  $q$ -eigenspace of Frobenius respectively) then the denominator  $f_2$  and leading coefficient  $f_3$  in Algorithm 3 are of this form, hence the name denominator elimination.
- The loop length and number of additions in Miller’s algorithm depend on the bit-length of  $S$  and its Hamming weight. Therefore, it is convenient to choose  $S$  as small as possible and to have low Hamming weight.
- Pairing value compression, using trace or torus methods [22, 43].
- Speeding up the final exponentiation, by exploiting the special form of the integer  $d$  and/or using trace or torus methods [4, 22, 24]. For instance, it is not difficult to prove that  $r \mid \Phi_k(q)$ , so a final exponentiation of the form  $(q^k - 1)/r$  can be written as

$$\frac{q^k - 1}{r} = \frac{\Phi_k(q)}{r} \cdot \prod_{s|k, s < k} \Phi_s(q)$$

where exponentiation by the last factor can be computed extremely fast using  $q$ -th power Frobenius operations.

The techniques mentioned above give impressive results for pairing implementation and they generalise trivially to hyperelliptic curves of genus  $g \geq 2$ .

## 7 Degenerate divisors versus general divisors

In non-pairing cryptography it was noted by Katagi et al. [28, 29] that using degenerate divisors can give performance advantages. In [5, 15] it is explained how degenerate divisors can be used to speed up pairing-based cryptosystems. For example, in the Boneh-Franklin identity-based encryption scheme [6], one can choose  $D_{\text{pub}}$  to be degenerate (i.e., choose  $D_{\text{pub}} = (P) - (\infty)$  first and then set  $D = [s^{-1}]D_{\text{pub}}$ ). One can also choose  $H(ID)$  to be degenerate, so that we hash to points rather than general divisors. Encryption in the Boneh-Franklin scheme then involves a pairing of two degenerate divisors, so is fast. Decryption still requires pairings of general divisors.

Several practical issues arise. First, can one choose a divisor  $D_{\text{pub}}$  of prime order of the form  $(P) - (\infty)$ ? If one is choosing elements of  $G_1$  and the divisor class group has prime order then this is automatic. The general case is discussed by Frey and Lange [15]. They also note that, when  $k$  is even, one might choose the second pairing argument from

$$\{(x, y) \in \mathbb{F}_{q^k} : x \in \mathbb{F}_{q^{k/2}}, y \in \mathbb{F}_{q^k} \setminus \mathbb{F}_{q^{k/2}}\},$$

which is not a group but which is a much bigger set than the usual choice of  $G_2$ .

Second, is there a loss of security from using degenerate divisors? It is easy to show that the DLP for degenerate divisors is not easier than the general DLP [29]. However, some protocols require more for security than just hardness of the DLP. For example, in the Boneh-Franklin scheme it must be hard to find collisions  $H(ID_1) = H(ID_2)$ . The presentation in [5] is disingenuous in this regard: since the number of degenerate divisors is roughly  $q$ , one can find collisions in time  $O(q^{1/2})$ , yet [5] gives running times for examples where the security level is supposed to be  $O(q)$ .

To summarise, the three advantages of using degenerate divisors in pairing based cryptography are: faster computation of divisor multiplication and pairings, simpler hashing into the group, and reduced bandwidth for transmitting group elements. Only the first of these is really an advantage when compared with elliptic curves. Hashing for elliptic curves is always to a single point, so is easy (and we can often exploit twists to make it faster [26]). Regarding small representatives of group elements, for elliptic curves over  $\mathbb{F}_p$  one could use points  $(x, y) \in E(\mathbb{F}_p)$  where  $1 < x < p^{1/g}$  (with a single bit to determine  $y$ ) and so the bandwidth requirement is the same as the hyperelliptic case.

## 8 Torsion and distortion

The results of this section are taken from [20, 41]. Let  $\mathcal{C}$  be a supersingular curve of genus  $g$  over  $\mathbb{F}_q$  and let  $r$  be prime and coprime to  $q$ . It is a standard fact that  $\text{Pic}_{\mathbb{F}_q}^0(\mathcal{C})[r]$  is isomorphic to  $(\mathbb{Z}/r\mathbb{Z})^{2g}$ . We let  $e$  be any non-degenerate, bilinear, Galois-invariant pairing which maps to  $\mu_r$ .

For pairings on elliptic curves with  $k > 1$  one typically has  $\mathcal{E}(\overline{\mathbb{F}}_q)[r] = \langle P, Q \rangle$  (here the notation  $\langle P, Q \rangle$  means the subgroup generated by  $P$  and  $Q$ ) where  $P$  is defined over  $\mathbb{F}_q$  and  $Q$  is defined over  $\mathbb{F}_{q^k}$ . It follows that  $e(P, P) = 1$  and hence, by non-degeneracy,  $e(P, Q) \neq 1$ . In other words, it is relatively simple to classify pairs of points whose pairing is non-trivial. In particular, if  $P$  is  $\mathbb{F}_q$ -rational and  $\psi$  is any endomorphism of  $\mathcal{E}$  such that  $\psi(P) \notin \langle P \rangle$  then it follows that  $e(P, \psi(P)) \neq 1$ .

In the higher genus case things are more complicated. Since a pairing  $e(D, \cdot)$  defines a linear map from  $(\mathbb{Z}/r\mathbb{Z})^{2g}$  to  $\mathbb{Z}/r\mathbb{Z}$  it follows that the kernel of  $e(D, \cdot)$  is  $(2g - 1)$ -dimensional. Hence, the condition  $\psi(D) \notin \langle D \rangle$  is not sufficient to imply that  $e(D, \psi(D)) \neq 1$ .

**Definition 2.** ([20, 50, 51]) *A distortion map for a non-degenerate pairing  $e$  and non-zero divisor classes  $D_1, D_2$  of prime order  $r$  on  $\mathcal{C}$  is an endomorphism  $\psi$  of  $\text{Jac}(\mathcal{C})$  such that  $e(D_1, \psi(D_2)) \neq 1$ .*

It was proved in [51] that distortion maps always exist for supersingular elliptic curves over  $\mathbb{F}_q$ . The result was generalised in [20]. We denote by  $\text{End}(A)$  the ring of endomorphisms on the abelian variety  $A$  defined over  $\overline{\mathbb{F}}_q$ .

**Theorem 3.** ([20]) *Let  $A$  be a supersingular abelian variety of dimension  $g$  over  $\mathbb{F}_q$ . Let  $r \mid \#A(\mathbb{F}_q)$  be prime. Let  $D_1, D_2$  be non-trivial elements of  $A(\overline{\mathbb{F}}_q)$  of order  $r$ . Then there is an element  $\psi \in \text{End}(A)$  such that  $e(D_1, \psi(D_2)) \neq 1$ .*

Furthermore, it is shown in [20] that, to have distortion maps for every non-trivial pair of divisors, it is necessary that the  $\mathbb{Z}$ -rank of the endomorphism ring is equal to  $(2g)^2$ . In other words, if  $\text{Jac}(\mathcal{C})$  has endomorphism ring which has  $\mathbb{Z}$ -rank strictly less than  $(2g)^2$  then there will exist non-zero divisor classes  $D_1$  and  $D_2$  in  $\text{Jac}(\mathcal{C})[r]$  such that  $e(D_1, \psi(D_2)) = 1$  for all  $\psi \in \text{End}(\text{Jac}(\mathcal{C}))$ . In other words, if  $\mathcal{C}$  is not supersingular then there cannot be distortion maps for every pair  $(D_1, D_2)$ .

In practice, as in Section 4.2, one tends to choose divisors  $D_1$  and  $D_2$  which lie in eigenspaces of the  $q$ -power Frobenius  $\pi$ . The following results may be of practical relevance in this setting.

**Lemma 1.** *Let  $A$  be a supersingular abelian variety over  $\mathbb{F}_q$  with characteristic polynomial of Frobenius equal to  $T^4 + aT^2 + q^2$  and suppose  $r \mid \#A(\mathbb{F}_q) = (q^2 + a + 1)$ . Then the Frobenius eigenvalues on  $A[r]$  are  $1, -1, q, -q$ .*

*Proof.* We have  $a \equiv -(q^2 + 1) \pmod{r}$ . Hence,

$$\begin{aligned} (T-1)(T-q)(T+1)(T+q) &= (T^2 - (q+1)T + q)(T^2 + (q+1)T + q) \\ &= T^4 - (q^2 + 1)T^2 + q^2 \\ &\equiv T^4 + aT^2 + q^2 \pmod{r}. \end{aligned}$$

Since the splitting of the characteristic polynomial of Frobenius is of this form, then the eigenvalues of Frobenius on  $A[r]$  are  $(1, -1, q, -q)$ .  $\square$

**Lemma 2.** *With notation as above, let  $(D_1, D_2, D_3, D_4)$  be an ordered  $\pi$ -eigenbasis for  $A[r]$  with eigenvalues  $(1, -1, q, -q)$  respectively. Suppose  $r \nmid (q^2 - 1)$ . Then if  $1 \leq i, j \leq 4$ , we have  $e(D_i, D_j) = 1$  unless  $(i, j) = (1, 3), (3, 1), (2, 4)$  or  $(4, 2)$ .*

*Proof.* We use Galois invariance of  $e$ . For example, for  $D_1$  one has

$$\pi(e(D_1, D_1)) = e(\pi(D_1), \pi(D_1)) = e(D_1, D_1).$$

This implies  $e(D_1, D_1) \in \mathbb{F}_q \cap \mu_r$  (recall that  $\mu_r$  is the group of  $r$ -th roots of unity and  $r \nmid (q-1)$ ) and hence  $e(D_1, D_1) = 1$ .

Similarly,

$$e(D_1, D_2)^q = \pi(e(D_1, D_2)) = e(\pi(D_1), \pi(D_2)) = e(D_1, -D_2) = e(D_1, D_2)^{-1}.$$

Since,  $r \nmid (q+1)$  this implies  $e(D_1, D_2) = 1$ .

Similarly,

$$e(D_1, D_4)^q = \pi(e(D_1, D_4)) = e(\pi(D_1), \pi(D_4)) = e(D_1, -qD_4) = e(D_1, D_4)^{-q}.$$

Since  $r \nmid 2q$  it follows that  $e(D_1, D_4) = 1$ . By non-degeneracy of  $e$ , one must have  $e(D_1, D_3) \neq 1$ .

The other cases are similar.  $\square$

It would be interesting to develop cryptographic protocols which utilise this torsion structure and the properties of pairings stated in the above Lemma.

We see that  $\pi$  can be used as a distortion map. For example, suppose  $D = D_1 + D_2$  and  $D' = D_3 + mD_4$ , with respect to the basis above, where  $m \in \mathbb{Z}$  is such that  $e(D, D') = e(D_1, D_3)e(D_2, D_4)^m = 1$ . Then we have

$$e(D, \pi(D')) = e(D_1, qD_3)e(D_2, -qmD_4) = e(D_1, D_3)^q e(D_2, D_4)^{-qm}$$

and this is not equal to 1 if  $m \not\equiv 0 \pmod{r}$ . Note that, for efficient implementation, there are often reasons to prefer the trace map  $\text{Tr}(D) = \sum_{i=0}^3 \pi^i(D)$  to  $\pi$ , though in the above example we have  $\text{Tr}(D') = 0$  so in this particular case the trace map is not useful.

## 9 Rubin-Silverberg point compression

Rubin and Silverberg [42] (also see [44, 45, 48]) have proposed an alternative way to view pairings on abelian varieties. They observe that many supersingular abelian varieties can be identified with subvarieties of Weil restrictions of supersingular elliptic curves. An alternative way to view their method is as a form of point compression for elliptic curves.

For example, the supersingular genus 2 curve  $\mathcal{C}$  over  $\mathbb{F}_2$  considered in [5] (also see Section 5.2) has  $k = 12$ . Working over  $\mathbb{F}_{2^m}$ , where  $m$  is coprime to 12, the number of points on the Jacobian of  $\mathcal{C}$  is  $N = 2^{2m} \pm 2^{(3m+1)/2} + 2^m \pm 2^{(m+1)/2} + 1$ . As mentioned above, one can use this curve for pairing-based cryptography and the finite field security comes from  $\mathbb{F}_{2^{12m}}$ . Alternatively, one can consider the supersingular elliptic curve  $E_b : y^2 + y = x^3 + x + b$  with  $b = 0, 1$  over  $\mathbb{F}_{2^{3m}}$ . This curve has  $k = 4$ , so we also map into  $\mathbb{F}_{2^{12m}}$ . Furthermore, the order of  $E_b(\mathbb{F}_{2^{3m}})$  (for the right choice of  $b$ ) is divisible by  $N$ . Indeed,  $\text{Jac}(\mathcal{C})$  can be identified with the trace zero part of the Weil restriction of scalars  $E_b$  with respect to  $\mathbb{F}_{2^{3m}}/\mathbb{F}_{2^m}$ .

In the above example, from a security point of view, there is no difference between computing pairings on  $E_b(\mathbb{F}_{2^{3m}})$  and  $\text{Jac}(\mathcal{C})(\mathbb{F}_{2^m})$ . However, one can represent a general divisor on the genus 2 curve over  $\mathbb{F}_{2^m}$  with about  $2m$  bits, whereas it requires about  $3m$  bits to represent a point in  $E_b(\mathbb{F}_{2^{3m}})$ . The contribution of Rubin and Silverberg is to give a method to represent elements of order  $N$  in  $E_b(\mathbb{F}_{2^{3m}})$  using only  $2m$  bits. In other words, the ‘‘per bit’’ security of the genus 2 curve is attained using elliptic curves. For precise details in this case see [5, 48].

The above is an example of the Rubin-Silverberg method with, in their notation,  $r = 3$ . The compression method is trivial (and fast). The decompression method involves solving some non-linear equations over the finite field, and it is practical only for small values of  $r$  (e.g.,  $r = 3$  or 5). Note that the method can be used for any elliptic curve, not just supersingular ones.

In [5] a performance comparison is given for the above example. When pairing degenerate divisors on the genus 2 curve (such elements do not correspond to special points on the elliptic curve side) the timings in [5] show that using genus 2 curves in this setting is faster than the Rubin-Silverberg approach. However, if

required to compute the pairing of general divisors on the genus 2 curve, then the timings indicate that the Rubin-Silverberg approach using the eta /ate pairing on supersingular elliptic curves is faster.

It therefore seems that, for parameters of current practical interest, it is more efficient to use elliptic curves with the Rubin-Silverberg compression method, than to work with divisor class groups of curves of genus  $g \geq 2$ . However, it should be noted that there are three possible exceptions to this statement: when degenerate divisors can be exploited the pairing may be faster for curves of genus  $g \geq 2$ ; the decompression method is only practical for small values of  $r$  (whereas comparable performance with some high genus curves could require larger  $r$ ); Abelian varieties have a richer torsion structure which may be useful for some applications.

## 10 Comparison of pairings on elliptic and hyperelliptic curves

In this section we compare the characteristics of pairings on elliptic and hyperelliptic curves of genus  $g$ . We write

$$e : G_1 \times G_2 \rightarrow G_T \subseteq \mathbb{F}_{q^k}^\times$$

for any of the pairings considered above. Here we assume that  $G_1$  is the subgroup defined over the small field (so it has the more compact representation) and  $G_2$  is the subgroup potentially defined over a larger field (hence, if using the ate pairing then  $e(P, Q) = a_S(Q, P)$ ).

The main criteria for our comparison are:

- Computation time (for operations in  $G_1, G_2, G_T$  as well as for computing  $e$ ).
- Size of representation (for  $G_1, G_2$  and  $G_T$ ).
- Flexibility and efficiency of parameter generation.
- Any other special properties.

### 10.1 Computation time

In most situations we may assume that  $G_1$  is a group of prime order  $r$ , consisting of elements defined over  $\mathbb{F}_q$ , with  $r \approx q^g$ . In this case, according to [33, 34], hyperelliptic curves can be as much as twice as fast as elliptic curves. If the hyperelliptic curve is more general (e.g., has 2 points at infinity) then the computation times in  $G_1$  may be a little slower for the hyperelliptic curve than the elliptic case.

If  $G_1$  is a rather small subgroup of the whole curve group then define  $\rho = g \log(q) / \log(r)$ . If the elliptic and hyperelliptic cases have similar values for  $\rho$  then the performance should be comparable. But operations on a hyperelliptic curve with  $\rho \geq 2$  (for example, Freeman's genus 2 curves [14]) will be slower than an elliptic curve with  $\rho \approx 1$ .

We now consider operations in  $G_2$ . In the supersingular case,  $G_2 = G_1$  and the above remarks apply. More generally, we aim to take  $G_2$  to be a subgroup of the divisor class group of a twist of the curve. The field of definition of  $G_2$  depends on the field  $\mathbb{F}_{q^k}$  (i.e., it is a function of the embedding degree). The crucial observation is that the field  $\mathbb{F}_{q^k}$  is required to be the same size regardless of the genus. Hence, the group  $G_2$  will be defined over a finite field of size independent of the genus. We therefore expect that operations in  $G_2$  will be slower in the hyperelliptic case than in the elliptic case, unless one can exploit twists of high degree (e.g., degree  $> 6g$ ).

The field  $\mathbb{F}_{q^k}$  is the same size whether using elliptic or hyperelliptic curves. So the computation time in  $G_T$  is the same in both cases.

We now consider the cost of computing pairings. First we present an oversimplified analysis: the dominant part of a standard Tate-Lichtenbaum pairing computation (on either an elliptic or hyperelliptic curve) is  $\log_2(r)$  iterations of computing  $h(D)$  for some function  $h$  and some divisor  $D$  (and accumulating the product of these values). As mentioned above, the field of definition of  $D$  depends on the embedding degree and so is similar in both cases. Furthermore, the functions  $h$  are more complicated for hyperelliptic curves than for elliptic curves. Hence, in general we expect pairing computation on hyperelliptic curves to be slower than elliptic curves.

The above analysis is extremely oversimplified and, indeed, is contradicted by the fact that the fastest known pairing computation is the  $\eta_T$  pairing on a supersingular genus 2 curve in characteristic 2 with embedding degree 12 (see the timings in Section 10 of [5]). The speed in that case is due to the implementation tricks available (including exploiting features of the processor architecture which are favourable to the genus 2 case). Also, the quoted timing is when using degenerate divisors; if general divisors are used then one gets faster timings using elliptic curves.

With hyperelliptic ate pairings over  $\mathbb{F}_q$  one has a loop of length approximately  $\log_2(q)$  in genus  $g$  rather than  $g \log_2(q)$ . This can be matched using the elliptic ate pairing if one can construct an elliptic curve over  $\mathbb{F}_p$  (for  $\log_2(p) \geq g \log_2(q)$ ) with trace of Frobenius  $t \approx q$ . The Brezing-Weng method [8] can be used to construct such curves. The example in Section 5.1 with  $n = 1$  gives  $\log_2(t) \approx \log_2(r)/4$ , which matches the loop shortening obtained by using genus 4 curves. Hence, despite the attractive properties of the hyperelliptic ate pairing, it seems that in practice one can always match the speed of pairing computation by using elliptic curves.

## 10.2 Size of representation

In general, we expect the size of the representation of  $G_1$  to be the same for both elliptic and hyperelliptic curves (at least, as long as the  $\rho$  values are comparable in both cases). As noted earlier, degenerate divisors in  $G_1$  require less storage than general elements, but there are potential security issues to take into account here. Also, as mentioned, the same effect can be achieved using elliptic curves by selecting points with short representations.

One issue here is comparing differing embedding degrees. For example, with supersingular curves in characteristic 2 we have  $k = 4$  for elliptic curves and  $k = 12$  for genus 2. One therefore would expect more compact representations in genus 2. However, the Rubin-Silverberg point compression method for elliptic curves can be applied. Hence, in practice, it seems that the size of elements of  $G_1$  is always no worse for elliptic curves than hyperelliptic curves.

We now consider the size of elements of  $G_2$ . In the supersingular case,  $G_1 = G_2$  and so the above remarks apply. If not, as mentioned above, the field of definition of  $G_2$  depends on the embedding degree and so we expect the representation of  $G_2$  to be larger in the hyperelliptic case than the elliptic case.

The representation of  $G_T$  is usually the same in both cases. Trace or torus methods can be used to compress these values [22, 43, 44].

### 10.3 Flexibility and efficiency of parameter generation

The main concerns in this section are as follows. Are there pairing-friendly curves with useful parameters for secure and efficient implementation? Are there many possible examples or just a few? Is it easy to construct equations for such curves? Does the user have very fine control over the parameters?

If supersingular curves are used then there is roughly the same amount of flexibility in parameter generation for both elliptic and hyperelliptic curves.

There have been a number of results on constructing ordinary pairing-friendly elliptic curves using the CM method (see [13] for a survey). The outcome of this research is that there are many polynomial families of suitable curves. Further, one can generate curves with relatively small values of  $t$ , which are attractive due to the elliptic ate pairing [26]. So there is plenty of flexibility when choosing elliptic curves for pairing-based cryptography. In genus  $g \geq 2$  the situation is much less satisfactory as was discussed in Section 5.2.

### 10.4 Special properties

For pairings on elliptic curves there are great efficiency savings for computations in  $G_2$  (including hashing to  $G_2$ ) by using twists [26]. If non-supersingular hyperelliptic curves are to be competitive with ordinary elliptic curves then it is likely that similar techniques would have to be developed.

As noted in Section 8, in genus  $\geq 2$  there are larger torsion structures available and there is interesting pairing behaviour (see Lemma 2). It is natural to ask whether there might be novel cryptosystems which exploit this structure. Such applications would give renewed motivation for using hyperelliptic curves in pairing-based cryptography.

## 11 Conclusions and open problems

For non-pairing cryptography there are potential advantages of using hyperelliptic curves of genus  $g \geq 2$ . It is thus natural to consider hyperelliptic curves

for pairing-based cryptography. We have surveyed work in this area. Our analysis indicates that, in practice, hyperelliptic curves are not more efficient than elliptic curves for general pairing applications. The only potentially significant advantage of hyperelliptic curves in pairing-based cryptosystems seems to be the speed of operations in  $G_1$ . Hence, hyperelliptic curves may be preferable for protocols with few pairing computations but many operations in  $G_1$ .

We conclude with a list of open problems.

- Can further loop shortening (as in [26, 37]) be performed for the hyperelliptic ate pairing?
- A major problem is to give methods to construct non-supersingular pairing friendly curves of genus  $g \geq 2$  and  $k$  in the range, say,  $6g \leq k \leq 30g$ . Ideally, these curves would have a single point at infinity and would have useful twists (as in [26]).
- Give fast methods to compute pairings on hyperelliptic curves with two points at infinity [21] or on non-hyperelliptic curves.
- Consider whether efficient and secure pairing-based cryptosystems can be developed for curves of genus  $g \geq 3$ , in spite of the index calculus attacks on curves in this case.
- Exploit the richer torsion structure available for abelian varieties. In particular, find cryptographic applications of pairings on groups which require 3 or more generators.  
A related problem is to give efficient methods to choose divisors in the particular subgroups.
- Improve the efficiency of the Rubin-Silverberg elliptic curve point decomposition method. Generalise the Rubin-Silverberg method to divisor class groups of curves of genus  $g \geq 2$ .
- In Section 9 we recalled the identification of certain abelian varieties with subvarieties of the Weil restriction of supersingular curves. In the case where the abelian variety is a Jacobian, is there a way to compute explicit homomorphisms between the elliptic curve representation and the Jacobian representation?

## Acknowledgements

We thank Tanja Lange, Eunjeong Lee, Jordi Pujolas, Mike Scott and Alice Silverberg for comments on a draft of this article.

The work described in this paper has been supported in part by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT. The first author also thanks the EPSRC for support.

## References

1. R. Avanzi, H. Cohen, C. Doche, G. Frey, T. Lange, K. Nguyen, and F. Vercauteren. *Handbook of elliptic and hyperelliptic curve cryptography*. Discrete Mathematics and its Applications. Chapman & Hall/CRC, 2006.

2. P. S. L. M. Barreto, H. Y. Kim, B. Lynn, and M. Scott. Efficient algorithms for pairing-based cryptosystems. In *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 354–368. Springer-Verlag, 2002.
3. P. S. L. M. Barreto, B. Lynn, and M. Scott. Constructing elliptic curves with prescribed embedding degrees. in S. Cimato et al (eds.), *SCN 2002*, volume 2576 of *Lecture Notes in Computer Science*, pages 257–267, Springer-Verlag 2003.
4. P. S. L. M. Barreto, B. Lynn, and M. Scott. Efficient implementation of pairing-based cryptosystems. *Journal of Cryptology*, 17(4):321–334, 2004.
5. P. S. L. M. Barreto, S. D. Galbraith, C. Ó hÉigearthaigh and M. Scott. Efficient pairing computation on supersingular Abelian varieties. *Designs, Codes and Cryptography*, **42**, No. 3 (2007) 239–271.
6. D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. *SIAM Journal of Computing*, 32(3):586–615, 2003.
7. D. Bernstein. Elliptic vs hyperelliptic, part 1. Talk at ECC 2006.
8. F. Brezing and A. Weng. Elliptic curves suitable for pairing based cryptography. *Designs, Codes and Cryptography*, 37, 133–141, 2005.
9. D. G. Cantor. Computing in the Jacobian of a hyperelliptic curve. *Math. Comp.*, 48(177):95–101, 1987.
10. Y.-J. Choie and E. Lee. Implementation of Tate pairing on hyperelliptic curves of genus 2. In *ICISC 2003*, volume 2971 of *Lecture Notes in Computer Science*, pages 97–111. Springer-Verlag, 2004.
11. I. Duursma and H.-S. Lee. Tate pairing implementation for hyperelliptic curves  $y^2 = x^p - x + d$ . In *Advances in Cryptology – ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 111–123. Springer-Verlag, 2003.
12. K. Eisentraeger, K. Lauter and P.L. Montgomery. Improved Weil and Tate pairings for elliptic and hyperelliptic curves. In *Algorithmic Number Theory - ANTS-VI*, volume 3076 of *Lecture Notes in Computer Science*, pages 169–183. Springer-Verlag 2004.
13. D. Freeman, M. Scott and E. Teske. A taxonomy of pairing-friendly elliptic curves. *Cryptology ePrint Archive*, Report 2006/372, 2006. Available from <http://eprint.iacr.org/2006/372>.
14. D. Freeman. Constructing pairing-friendly genus 2 curves over prime fields with ordinary Jacobians. To appear in proceedings of Pairing 2007.
15. G. Frey and T. Lange. Fast bilinear maps from the Tate-Lichtenbaum pairing on hyperelliptic curves. In *Algorithmic Number Theory – ANTS VII*, volume 4076 of *Lecture Notes in Computer Science*, pages 466–479. Springer-Verlag, 2006.
16. G. Frey and H.-G. Rück. A remark concerning  $m$ -divisibility and the discrete logarithm problem in the divisor class group of curves. *Math. Comp.*, 52:865–874, 1994.
17. S. D. Galbraith, K. Harrison, and D. Soldera. Implementing the Tate pairing. In *Algorithmic Number Theory – ANTS V*, volume 2369 of *Lecture Notes in Computer Science*, pages 324–337. Springer-Verlag, 2002.
18. S. D. Galbraith. Supersingular curves in cryptography. In *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 495–513. Springer-Verlag, 2001.
19. S. D. Galbraith, J. F. McKee and P. C. Valença, Ordinary abelian varieties having small embedding degree. *Finite Fields and Their Applications*, doi:10.1016/j.ffa.2007.02.003.
20. S. D. Galbraith, J. Pujolàs, C. Ritzenthaler and B. Smith. Distortion maps for genus two curves, preprint, [arXiv:math/0611471](https://arxiv.org/abs/math/0611471) (2006).

21. S. D. Galbraith and D. Mireles. Computing pairings on general genus 2 curves, preprint (2007).
22. R. Granger, D. Page, and M. Stam. On small characteristic algebraic tori in pairing-based cryptography. In *LMS Journal of Computation and Mathematics*, volume 9, pages 64–85, 2006.
23. R. Granger, F. Hess, R. Oyono, N. Thériault and F. Vercauteren. ate pairing on hyperelliptic curves. In *Advances in Cryptology – EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 419–436. Springer-Verlag, 2007.
24. R. Granger, D. Page and N. P. Smart. High security pairing-based cryptography revisited. In *Algorithmic Number Theory – ANTS VII*, volume 4076 of *Lecture Notes in Computer Science*, pages 480–494. Springer-Verlag, 2006.
25. R. Harley. Fast arithmetic on genus 2 curves. Available at <http://cristal.inria.fr/~harley/hyper>, 2000.
26. F. Hess, N. P. Smart and F. Vercauteren, The Eta Pairing Revisited. *IEEE Trans. Information Theory*, 52(10): 4595–4602, 2006.
27. A. Joux. A one round protocol for tripartite Diffie-Hellman. *Journal of Cryptology*, 17 (4) : 263–276, 2004.
28. M. Katagi, T. Akishita, I. Kitamura, and T. Takagi. Some improved algorithms for hyperelliptic curve cryptosystems using degenerate divisors. In *ICISC 2004*, volume 3506 of *Lecture Notes in Computer Science*, pages 296–312. Springer-Verlag, 2005.
29. M. Katagi, I. Kitamura, T. Akishita, and T. Takagi. Novel efficient implementations of hyperelliptic curve cryptosystems using degenerate divisors. In *Information Security Applications – WISA ’2004*, volume 3325 of *Lecture Notes in Computer Science*, pages 345–359. Springer-Verlag, 2005.
30. N. Koblitz. Hyperelliptic cryptosystems. *Journal of Cryptology*, 1(3):139–150, 1989.
31. N. Koblitz and A. Menezes. Pairing-based cryptography at high security levels. In *Cryptography and Coding – Cirencester 2005*, volume 3796 of *Lecture Notes in Computer Science*, pages 13–36. Springer-Verlag, 2005.
32. T. Lange. Formulae for arithmetic on genus 2 hyperelliptic curves. In *Appl. Algebra Eng. Commun. Comput.*, 15(5):295–328, 2005.
33. T. Lange and M. Stevens. Efficient doubling on genus two curves over binary fields. In *Selected Areas in Cryptography – SAC 2004*, volume 3357 of *Lecture Notes in Computer Science*, pages 170–181. Springer-Verlag, 2004.
34. T. Lange. Elliptic vs. hyperelliptic, part 2. Talk at ECC 2006.
35. E. Lee, H.-S. Lee and Y. Lee. Eta pairing computation on general divisors over hyperelliptic curves  $y^2 = x^7 - x \pm 1$ . To appear in proceedings of Pairing 2007.
36. S. Lichtenbaum. Duality theorems for curves over  $p$ -adic fields. *Invent. Math.*, 7:120–136, 1969.
37. S. Matsuda, N. Kanayama, F. Hess, E. Okamoto. Optimized versions of the ate and twisted ate pairings. Cryptology ePrint Archive, Report 2007/013, 2007. Available from <http://eprint.iacr.org/2007/013>.
38. V. S. Miller. The Weil pairing and its efficient calculation. *Journal of Cryptology*, 17(4):235–261, 2004.
39. D. Mumford. Abelian Varieties. Oxford University Press, London (1970).
40. C. Ó hÉigartaigh and M. Scott. Pairing calculation on supersingular genus 2 curves. To appear in Selected Areas in Cryptography, 2006.
41. J. Pujolas. On the decisional Diffie-Hellman problem in genus 2, PhD thesis, Universitat Politècnica de Catalunya, 2006.

42. K. Rubin and A. Silverberg. Supersingular abelian varieties in cryptology. In *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 336–353. Springer-Verlag, 2002.
43. K. Rubin and A. Silverberg. Torus-based cryptography. In *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 349–365. Springer-Verlag, 2003.
44. K. Rubin and A. Silverberg. Using primitive subgroups to do more with fewer bits. In *Algorithmic Number Theory – ANTS VI*, volume 3076 of *Lecture Notes in Computer Science*, pages 18–41. Springer-Verlag, 2004.
45. K. Rubin and A. Silverberg. Using abelian varieties to improve pairing based cryptography. Preprint 2007.
46. R. Sakai, K. Ohgishi and M. Kasahara. Cryptosystems based on pairing. In *The 2000 Symposium on Cryptography and Information Security (SCIS 2000)*, Okinawa, Japan, January 2000.
47. R. Sakai, K. Ohgishi and M. Kasahara. Cryptosystems based on pairing over elliptic curve (in Japanese). In *The 2001 Symposium on Cryptography and Information Security*, Oiso, Japan, January 2001.
48. A. Silverberg. Compression for trace zero subgroups of elliptic curves. *Trends in Mathematics*, **8** (2005) 93–100.
49. J. Tate. WC group over  $\mathfrak{p}$ -adic fields. *Séminaire Bourbaki*, 1958.
50. E. Verheul, Evidence that XTR is more secure than supersingular elliptic curve cryptosystems. In *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 195–210. Springer-Verlag, 2001.
51. E. Verheul, Evidence that XTR is more secure than supersingular elliptic curve cryptosystems, *Journal of Cryptology*, 17, No. 4 (2004) 277–296.
52. A. Weil. Sur les fonctions algébriques à corps de constantes finis. C. R. Acad. Sci. Paris, 210:592-594, 1940 (= Oeuvres Scientifiques, Volume I, pp. 257-259).