Endomorphisms for faster elliptic curve cryptography on general curves

Steven D. Galbraith*, Xibin Lin**, and Michael Scott***

[1] Mathematics Department,
Royal Holloway, University of London,
Egham, Surrey, TW20 0EX,
United Kingdom.
steven.galbraith@rhul.ac.uk
[2] School of Mathematics and Computational Science
Sun-Yat Sen University Guangzhou, 510275, P.R.China
linxibin@mail2.sysu.edu.cn
[3] School of Computing, Dublin City University.
Ballymun, Dublin 9, Ireland.
mike@computing.dcu.ie

Abstract. Efficiently computable homomorphisms allow multiplication to be accelerated using the Gallant-Lambert-Vanstone (GLV) method. We extend results of Iijima, Matsuo, Chao and Tsujii which give such homomorphisms for general elliptic curves by working over quadratic extensions and demonstrate these results can be applied to the GLV method. Our preliminary results give up to a 72 percent speedup (i.e., 0.58 times the previous best time) for elliptic curve point multiplication on curves without small class number complex multiplication. Further speedups are possible when using special curves.
Keywords: elliptic curves, point multiplication, GLV method, isogenies.

1 Introduction

Let E be an elliptic curve over a finite field $\mathbb{F}_q$ and let $P \in E(\mathbb{F}_q)$ have order r. The fundamental operation in elliptic and hyperelliptic curve cryptography is point multiplication $[n]P$ where $n \in \mathbb{Z}$. There is a vast literature on efficient methods for computing $[n]P$ (a good reference is [2]).

The Gallant-Lambert-Vanstone method [12] is an important tool for speeding up point multiplication. The basic idea is as follows. If the elliptic curve E has a efficiently computable endomorphism ψ (other than a standard multiplication by n map) such that $\psi(P) \in \langle P \rangle$ then one can replace the computation $[n]P$ by the multiexponentiation $[n_0]P + [n_1]\psi(P)$ where $|n_0|, |n_1| \approx \sqrt{r}$. In principle this makes the computation $[n]P$ nearly twice as fast. In practice the speedup is not 100 percent, but is still very significant. Some examples allow higher degree decompositions such as $[n_0] + [n_1]\psi(P) + \cdots + [n_{m-1}]\psi^{m-1}(P)$ where $|n_i| \approx r^{1/m}$ which gives further speedups (though, for technical reasons discussed in Section 6, one does not expect it to be m-times faster than the other best methods). We call the latter approach the m-dimensional GLV method.

Gallant, Lambert and Vanstone [12] only gave examples of suitable efficiently computable endomorphisms in two cases, namely subfield curves (i.e., groups $E(\mathbb{F}_{q^m})$ where E is defined over

* This work supported by EPSRC grant EP/D069904/1.
** This author thanks the Chinese Scholarship Council.
*** This author acknowledges support from the Science Foundation Ireland under Grant No. 06/MI/006

$\mathbb{F}_q$; these do not have prime or nearly prime order unless q is very small) and curves with special endomorphism structure (essentially, that the endomorphism ring has small class number). Hence, if one is using randomly chosen prime-order elliptic curves over finite fields for cryptography then the GLV method is not usually available. Indeed, in Section 7 of [24] one finds the claim "the GLV method is only effective for those exceptional elliptic curves that have complex multiplication by an order with small discriminant."

In fact, Iijima, Matsuo, Chao and Tsujii [17] constructed an efficiently computable homomorphism on elliptic curves $E(\mathbb{F}_{p^2})$ with $j(E) \in \mathbb{F}_p$ arising from the Frobenius map on a twist of E. Apparently they did not realise the application of their results to the GLV method. In this paper we give a generalisation of this approach and analyse it in the context of the GLV method. The techniques apply to all elliptic curves over $\mathbb{F}_{p^2}$ such that $j(E) \in \mathbb{F}_p$ (regardless of the endomorphism structure) and can be used with curves of prime order. Hence, although our curves are not completely general, we overturn the claims of Section 7 of [24].

The basic idea is somewhat analogous to subfield curves: We take elliptic curves E with $j(E) \in \mathbb{F}_q$ and consider the group $E(\mathbb{F}_{q^m})$. However a crucial difference is that E is defined over $\mathbb{F}_{q^m}$, not $\mathbb{F}_q$. This means it is possible to obtain curves of prime order and so there is no need to restrict attention to q being small. Our method can be used with any primes p and any elliptic curves E over $\mathbb{F}_q$ and always gives rise to a GLV method of dimension at least 2.

Our experimental results demonstrate up to a 72 percent speedup (i.e., 0.58 the time) over previous methods for point multiplication $[n](x, y)$ on general curves. Note that for some applications one could already obtain faster timings by using only x-coordinate arithmetic (i.e., Montgomery form); for examples see Bernstein [3] and Gaudry-Thomé [14]. These ideas can also be used in our setting, but with less of a speedup. Nevertheless, our methods can be applied in situations (such as signature verification) which are not covered by the timings in [3, 14] and we expect a significant speedup in this case.

Note that our techniques can also be implemented on curves in Edwards form, and exploit their benefits. We also stress that our approach requires no special properties of the elliptic curves E (except that $j(E) \in \mathbb{F}_p$) and so can be applied to elliptic curves of prime order.

We now give an outline of the paper. First we describe the homomorphism and explain how it leads to a 2-dimensional GLV method for any elliptic curve. Section 3 gives a specific key generation algorithm which may be convenient for some applications. Section 4 shows how to get a 4-dimensional GLV method for $y^2 = x^3 + B$ over $\mathbb{F}_{p^2}$. Section 5 shows that our homomorphisms can also be used for curves written in Edwards form. Section 8 discusses the generalisation of our methods to higher genus curves. The proof of the pudding is the timings in Section 9. Section 10 discusses known security threats from using our construction and explains how to avoid them.

2 The homomorphism

We consider elliptic curves defined over any field $\mathbb{F}_q$ with point at infinity ∞. Recall that if E is an elliptic curve over $\mathbb{F}_q$ with $q + 1 - t$ points then one can compute the number of points $\#E(\mathbb{F}_{q^m})$ efficiently. For example, $\#E(\mathbb{F}_{q^2}) = q^2 + 1 - (t^2 - 2q) = (q + 1)^2 - t^2$. As usual we define

$$E(\mathbb{F}_{q^m})[r] = \{P \in E(\mathbb{F}_{q^m}) : [r]P = \infty\}.$$

When we say that a curve or mapping is 'defined over $\mathbb{F}_{q^k}$' we mean that the coefficients of the polynomials are all in $\mathbb{F}_{q^k}$. The implicit assumption throughout the paper is that when we say

an object is defined over a field $\mathbb{F}_{q^k}$ then it is not defined over any smaller field, unless explicitly mentioned.

The following result gives our main construction. Novices can replace the word 'isogeny' with 'isomorphism', set $d = 1$ and replace $\hat{\phi}$ by ϕ^{-1} without any significant loss of functionality.

Theorem 1. *Let E be an elliptic curve defined over $\mathbb{F}_q$ such that $\#E(\mathbb{F}_q) = q + 1 - t$ and let $\phi : E \to E'$ be a separable isogeny of degree d defined over $\mathbb{F}_{q^k}$ where E' is an elliptic curve defined over $\mathbb{F}_{q^m}$ with $m \mid k$. Let $r \mid \#E'(\mathbb{F}_{q^m})$ be a prime such that $r > d$ and such that $r \| \#E'(\mathbb{F}_{q^k})$. Let π be the q-power Frobenius map on E and $\hat{\phi} : E' \to E$ be the dual isogeny of ϕ. Let*

$$\psi = \phi\pi\hat{\phi}.$$

Then

1. *$\psi \in End(E')$ (i.e., ψ is a group homomorphism).*
2. *For all $P \in E'(\mathbb{F}_{q^k})$ we have $\psi^k(P) - [d^k]P = \infty$ and $\psi^2(P) - [dt]\psi(P) + [d^2q]P = \infty$.*
3. *There is some $\lambda \in \mathbb{Z}$ such that $\lambda^k - d^k \equiv 0 \pmod{r}$ and $\lambda^2 - dt\lambda + d^2q \equiv 0 \pmod{r}$ such that $\psi(P) = [\lambda]P$ for all $P \in E'(\mathbb{F}_{q^m})[r]$.*

Proof. First note that $\hat{\phi}$ is an isogeny from E' to E and is defined over $\mathbb{F}_{q^k}$, that π is an isogeny from E to itself defined over $\mathbb{F}_q$, and that ψ is an isogeny from E to E' defined over $\mathbb{F}_{q^k}$. Hence ψ is an isogeny of E' to itself, and is defined over $\mathbb{F}_{q^k}$ (or a subfield). Therefore, ψ is a group homomorphism.

Since $\phi\hat{\phi} = d$ on E' it follows that

$$\psi^2 = \phi\pi\hat{\phi}\phi\pi\hat{\phi} = \phi\pi d\pi\hat{\phi} = d\phi\pi^2\hat{\phi}$$

and, by induction, $\psi^k = d^{k-1}\phi\pi^k\hat{\phi}$. For $P \in E'(\mathbb{F}_{q^k})$ we have $\hat{\phi}(P) \in E(\mathbb{F}_{q^k})$ and so $\pi^k(\hat{\phi}(P)) = \hat{\phi}(P)$. Hence $\psi^k(P) = [d^k]P$.

Similarly, writing $Q = \hat{\phi}(P)$ we have $\pi^2(Q) - [t]\pi(Q) + [q]Q = \infty$ and so $[d]\phi(\pi^2 - [t]\pi + [q])\hat{\phi}(P) = \infty$. Using the previous algebra, this implies

$$(\psi^2 - [dt]\psi + [qd^2])P = \infty.$$

Finally, let $P \in E'(\mathbb{F}_{q^m})$ have order r. Since $\psi(P) \in E'(\mathbb{F}_{q^k})$ also has order r and $r \| \#E'(\mathbb{F}_{q^k})$ it follows that $\psi(P) = [\lambda]P$ for some $\lambda \in \mathbb{Z}$. Clearly, $\psi([a]P) = [a]\psi(P) = [\lambda]([a]P)$ for all $a \in \mathbb{Z}$. Since $\psi^k(P) - [d^k]P = [\lambda^k]P - [d^k]P = \infty$ it follows that $\lambda^k - d^k \equiv 0 \pmod{r}$. Similarly, $\lambda^2 - dt\lambda + d^2q \equiv 0 \pmod{r}$. □

2.1 Special case of quadratic twists

We now give our general result, which applies to elliptic curves over $\mathbb{F}_p$. Our construction can of course be used for elliptic curves over fields of small characteristic, but it seems more natural to use subfield curves and Frobenius expansions in that case. Hence, for the remainder of the paper we focus on the case of characteristic $p > 3$.

Corollary 1. *Let $p > 3$ be a prime and let E be an elliptic curve over $\mathbb{F}_p$ with $p + 1 - t$ points. Let E' over $\mathbb{F}_{p^2}$ be the quadratic twist of $E(\mathbb{F}_{p^2})$, so that $\#E'(\mathbb{F}_{p^2}) = (p-1)^2 + t^2$. Write $\phi : E \to E'$ for the twisting isomorphism defined over $\mathbb{F}_{p^4}$. Let $r \mid \#E'(\mathbb{F}_{p^2})$ be a prime such that $r > 2p$ Let $\psi = \phi\pi\phi^{-1}$. For $P \in E'(\mathbb{F}_{p^2})[r]$ we have $\psi^2(P) + P = \infty$.*

Proof. Let $E : y^2 = x^3 + Ax + B$ with $A, B \in \mathbb{F}_p$. We have $\#E(\mathbb{F}_{p^2}) = p^2 + 1 - (t^2 - 2p)$. Let $u \in \mathbb{F}_{p^2}$ be a non-square in $\mathbb{F}_{p^2}$, define $A' = u^2A, B' = u^3B$ and $E' : y^2 = x^3 + A'x + B'$. Then $\#E'(\mathbb{F}_{p^2}) = p^2 + 1 + (t^2 - 2p) = (p-1)^2 + t^2$. The isomorphism $\phi : E \to E'$ is defined by

$$\phi(x, y) = (ux, \sqrt{u}^3 y)$$

and is defined over $\mathbb{F}_{p^4}$.

If $r \mid \#E'(\mathbb{F}_{p^2})$ is prime such that $r > 2p$ then $r \nmid \#E(\mathbb{F}_{p^2})$ and so $r \| \#E'(\mathbb{F}_{p^4}) = \#E(\mathbb{F}_{p^2})\#E'(\mathbb{F}_{p^2})$. Hence we may apply Theorem 1 to get that ψ is a group homomorphism such that $\psi(P) = \lambda P$ such that $\lambda^4 - 1 \equiv 0 \pmod{r}$ for $P \in E'(\mathbb{F}_{p^2})[r]$. We now show that, in fact, $\lambda^2 + 1 \equiv 0 \pmod{r}$.

By definition, $\psi(x, y) = (ux^p/u^p, \sqrt{u}^3 y^p / \sqrt{u}^{3p})$ where $u \in \mathbb{F}_{p^2}$ (i.e., $u^{p^2} = u$) and $\sqrt{u} \notin \mathbb{F}_{p^2}$ (and so, $\sqrt{u}^{p^2} = -\sqrt{u}$). If $P = (x, y) \in E'(\mathbb{F}_{p^2})$ then $x^{p^2} = x, y^{p^2} = y$ and so

$$\begin{aligned}\psi^2(x, y) &= (ux^{p^2}/u^{p^2}, \sqrt{u}^3 y^{p^2} / \sqrt{u}^{3p^2}) \\ &= (x, (-1)^3 y) \\ &= -P.\end{aligned}$$

This completes the proof. □

The above result applies to any elliptic curve over $\mathbb{F}_p$ (with $p > 3$) and shows that the 2-dimensional GLV method can be applied. Note that it is possible for $\#E'(\mathbb{F}_{p^2})$ to be prime, since E' is not defined over $\mathbb{F}_p$ (for further analysis see Nogami and Morikawa [20]). One feature of this construction is that, since p is now half the size compared with using elliptic curves over prime fields, point counting is much faster than usual (this was noted in [20]). Since we are dealing with elliptic curves over $\mathbb{F}_{p^2}$ where p is prime then Weil descent attacks are not a threat (see Section 10).

We stress that there is nothing unexpected in our construction. We know that $\mathrm{End}(E)$ contains the p-power Frobenius map and, since $E' \cong E$, we know that $\mathrm{End}(E') \cong \mathrm{End}(E)$ contains a corresponding endomorphism.

Corollary 2. *Let $p \equiv 5 \pmod{8}$ be a prime. Let notation be as in the previous corollary. Then one may choose*

$$\psi(x, y) = (-x^p, iy^p)$$

where $i \in \mathbb{F}_p$ satisfies $i^2 = -1$.

Proof. We have $4 \| (p-1)$ and $2 \| (p+1)$. Since 2 is not a square in $\mathbb{F}_p$ one can define $\mathbb{F}_{p^2} = \mathbb{F}_p(u)$ where $u = \sqrt{2}$. Note that $u^p = -u$ and that $u^{p-1} \equiv 2^{(p-1)/2} \equiv -1 \pmod{p}$. It follows that $u^{(p^2-1)/2} = -1$ and so u is not a square in $\mathbb{F}_{p^2}$.

Since -1 is a square in $\mathbb{F}_p$ the equation $x^4 = 1$ has solutions $x = 1, -1, i, -i \in \mathbb{F}_p$. Let $w \in \mathbb{F}_{p^4}$ satisfy $w^2 = u$. Note that $(w/w^p)^4 = 1$ and so $w^p = \pm iw$.

Finally, our homomorphism ψ is defined to be

$$\psi(x, y) = (ux^p/u^p, w^3 y^p / w^{3p}) = (-x^p, \pm iy^p).$$

Renaming i if necessary gives the result. □

An exercise for the reader is to show that if E is an elliptic curve over $\mathbb{F}_p$ and if E' over $\mathbb{F}_p$ is the quadratic twist of E then the map ψ of our construction satisfies $\psi(P) = -P$ for all $P \in E'(\mathbb{F}_p)$. Our homomorphism is therefore useless for the GLV method in this case.

2.2 Higher dimension decompositions

The GLV method can be generalised to m-dimensional decompositions $[n]P = [n_0]P + [n_1]\psi(P) + \cdots + [n_{m-1}]\psi^{m-1}(P)$ (for examples with $m = 4$ and $m = 8$ see [10]). Such a setting gives improved performance. As we have found 2-dimensional expansions using $E'(\mathbb{F}_{p^2})$ it is natural to try to get an m-dimensional decomposition using $E'(\mathbb{F}_{p^m})$.

In general, to obtain an m-dimensional decomposition it is required that ψ does not satisfy any polynomial equation on $E'(\mathbb{F}_{p^m})[r]$ of degree $< m$ with small integer coefficients. Note that ψ always satisfies a quadratic polynomial equation but that the coefficients are not necessarily small modulo r.

The following result gives a partial explanation of the behaviour of ψ on $E'(\mathbb{F}_{p^m})$.

Corollary 3. *Let $p > 3$ be a prime and let E be an elliptic curve over $\mathbb{F}_p$. Let E' over $\mathbb{F}_{p^m}$ be the quadratic twist of $E(\mathbb{F}_{p^m})$. Write $\phi : E \to E'$ for the twisting isomorphism defined over $\mathbb{F}_{p^{2m}}$. Let $r \mid \#E'(\mathbb{F}_{p^m})$ be a prime such that $r > 2p^{m/2}$ Let $\psi = \phi\pi\phi^{-1}$. For $P \in E'(\mathbb{F}_{p^m})[r]$ we have $\psi^m(P) + P = \infty$.*

Proof. As in Corollary 1, we have $r \| \#E'(\mathbb{F}_{p^{2m}}) = \#E'(\mathbb{F}_{p^m})\#E(\mathbb{F}_{p^m})$. Also, using the same method as the proof of Corollary 1 we have

$$\begin{aligned}\psi^m(x, y) &= (ux^{p^m}/u^{p^m}, \sqrt{u}^3 y^{p^m}/\sqrt{u}^{3p^m}) \\ &= -P.\end{aligned}$$

□

The problem is that the polynomial $x^m + 1$ is not usually irreducible, and it is possible that ψ satisfies a smaller degree polynomial. For example, in the case $m = 3$ one sees that $\#E'(\mathbb{F}_{p^3})$ cannot be prime as it is divisible by $N = \#E(\mathbb{F}_{p^2})/\#E(\mathbb{F}_p)$. If $r \mid \#E'(\mathbb{F}_{p^3})/N$ then $\psi^2(P) - \psi(P) + 1 = \infty$ for $P \in E'(\mathbb{F}_{p^3})[r]$. Hence one only gets a 2-dimensional decomposition in the case $m = 3$.

Indeed, the interesting case is when m is a power of 2, in which case $x^m + 1$ is irreducible and one can obtain an m-dimensional GLV decomposition. Indeed, Nogami and Morikawa [20] already proposed exactly this key generation method (choosing E over $\mathbb{F}_p$ and then using a quadratic twist over $\mathbb{F}_{p^{2^c}}$) as a method to generate curves of prime order. Note that [20] does not consider the GLV method.

Therefore, the next useful case is $m = 4$, giving a 4-dimensional GLV method. On the downside, this case is potentially vulnerable to Weil descent attacks (see Section 10) and so the prime p must be larger than we would ideally like.

The other way to get higher dimension decompositions is to have maps ϕ defined over larger fields than a quadratic extension. An example of this is given in Section 4.

3 Key generation

Let $p > 3$ be prime. We present a key generation algorithm for the construction based on quadratic twists. Our algorithm is designed so that the resulting curve $E' : y^2 = x^3 + A'x + B'$ over $\mathbb{F}_{p^2}$ has coefficient $A' = -3$, which is convenient for efficient implementation when using Jacobian coordinates (see Section 13.2 of [2]).

We follow the conditions of Corollary 2, which give a particularly simple map ψ. It should be clear that the algorithm can be used in more general cases.

1. Choose a prime $p = 5 \pmod 8$. Note that p can be a special prime, such as NIST prime (see Section 2.2.6 of [16]).
2. Set $u = \sqrt{2} \in \mathbb{F}_{p^2}$.
3. Set $A' = -3$ and $A = A'/2 \in \mathbb{F}_p$.
4. Repeat
   - Choose random $B \in \mathbb{F}_p$ and let $E : y^2 = x^3 + Ax + B$.
   - Compute $t = p + 1 - \#E(\mathbb{F}_p)$.
5. Until $(p-1)^2 + t^2 = hr$ where r is prime and $h = 1$ (or maybe $h < H$ for some bound H on the cofactor).
6. Set $B' = Bu^3 \in \mathbb{F}_{p^2}$ and $E' : y^2 = x^3 + A'x + B'$.
7. Compute $\lambda \in \mathbb{Z}$ such that $\lambda^2 + 1 \equiv 0 \pmod r$.
8. Compute $i \in \mathbb{F}_p$ so that $i^4 = 1$. Then $\psi(x, y) = (-x^p, iy^p)$.

One can directly verify that $(B')^p = -B'$ and so ψ clearly maps $E'(\mathbb{F}_{p^2})$ to itself. As in Corollary 1, the homomorphism ψ is can be used for a 2-dimensional GLV method, since $\psi(P) = \lambda P$ for $P \in E'(\mathbb{F}_{p^2})[r]$.

As remarked earlier, key generation is fast compared with standard ECC, since the point counting for $\#E(\mathbb{F}_p)$ is over a field half the usual size (this is precisely the point of the paper [20]).

4 Using special curves

We have seen that one can obtain a 2-dimensional GLV method for any elliptic curve over $\mathbb{F}_p$. However, 2-dimensional GLV methods were already known for some special curves (i.e., those with a non-trivial automorphism or endomorphism of low degree). We now show how one can get higher-dimensional expansions using elliptic curves E over $\mathbb{F}_{p^2}$ with $\#\mathrm{Aut}(E) > 2$.

The two examples of interest are $E : y^2 = x^3 + B$ and $y^2 = x^3 + Ax$. We give the details in the former case. The latter is analogous.

Let $p \equiv 1 \pmod 6$ and let $B \in \mathbb{F}_p$. Define $E : y^2 = x^3 + B$. Choose $u \in \mathbb{F}_{p^{12}}$ such that $u^6 \in \mathbb{F}_{p^2}$ and define $E' : Y^2 = X^3 + u^6 B$ over $\mathbb{F}_{p^2}$. Repeat the construction (choosing p, B, u) until $\#E'(\mathbb{F}_{p^2})$ is prime (or nearly prime). Note that there are 6 possible group orders for $y^2 = x^3 + B'$ over $\mathbb{F}_{p^2}$ and three of them are never prime as they correspond to group orders of curves defined over $\mathbb{F}_p$.

The isomorphism $\phi : E \to E'$ is given by $\phi(x, y) = (u^2 x, u^3 y)$ and is defined over $\mathbb{F}_{p^{12}}$. The homomorphism $\psi = \phi\pi\phi^{-1}$, where π is the p-power Frobenius on E, is defined over $\mathbb{F}_{p^2}$ and satisfies the characteristic equation

$$\psi^4 - \psi^2 + 1 = 0.$$

Hence one obtains a 4-dimensional GLV method for these curves. This leads, once again, to a nearly doubling of the speed of these curves compared with previous techniques.

5 Using Edwards curves

In this section we explain how to write our homomorphism in terms of the Edwards equation for an elliptic curve, under the assumption that our original curve E has a point of order 4 and a unique point of order 2 (these conditions imply that E can be written in Edwards form). This means that the multiexponentiation can be performed using the Edwards formulae for elliptic curve additions and doublings. We closely follow Section 2 of [6].

Let E be an elliptic curve $E/\mathbb{F}_{p^2} : y^2 = x^3 + Ax + B$ with a point $P = (r_1, s_1)$ of order 4 and a unique point $Q = [2]P = (r_2, 0)$ of order 2. Suppose we have an efficiently computable homomorphism $\psi : E(\mathbb{F}_{p^2}) \to E(\mathbb{F}_{p^2})$ of the form $\psi : (x, y) = (c_1 x^p, c_2 y^p)$, where $c_1, c_2 \in \mathbb{F}_{p^2}$ are constants. We assume that $\psi(P) = \lambda P$ for $P \in E(\mathbb{F}_{p^2})$ for some $\lambda \in \mathbb{Z}$. We will transform E to an Edwards elliptic curve E_e with an efficiently computable homomorphism ψ_e on it, such that $\psi_e(P) = [\lambda]P$ for $P \in E_e(\mathbb{F}_{p^2})$.

We first move Q to $(0, 0)$ using the standard transformation $\chi_1(x, y) = (x - r_2, y)$ which maps to $E_1 : y^2 = X^3 + a_2X^2 + a_4X$, where $a_2 = 3r_2$ and $a_4 = 3r_2^2 + A$ are defined over $\mathbb{F}_{p^2}$.

As in [6], let $d = 1 - \frac{4r_1^3}{s_1^2} \in \mathbb{F}_{p^2}$ and consider the elliptic curve $E_e : \bar{x}^2 + \bar{y}^2 = 1 + d\bar{x}^2\bar{y}^2$ in Edwards form. We now present the explicit birational map from $E_1 \to E_e$ given in [6].

Let $t_1 = \sqrt{\frac{r_1}{1-d}} = \pm\frac{s_1}{2r_1}$. Define $E_2 : (\frac{r_1}{1-d})y^2 = x^3 + a_2x^2 + a_4x$. Then there is an isomorphism χ_2 from $E_1 \to E_2$ by $\chi_2(x, y) = (x, y/t_1)$. As explained in [6], we know that $a_4 = r_1^2$ and $a_2 = 2r_1(1+d)/(1-d)$. Let $E_3 : (\frac{1}{1-d})y^2 = x^3 + 2((1+d)/(1-d))x^2 + x$. The isomorphism χ_3 from E_2 to E_3 is given by $\chi_3(x, y) = (x/r_1, y/r_1)$. Finally, E_3 is birationally equivalent to E_e by the birational map $\chi_4(x, y) = (2x/y, (x - 1)/(x + 1))$.

To summarize the above, there is a birational map ρ from E to E_e given by

$$\rho(x, y) = \left(\frac{2t_1(x - r_2)}{y}, \frac{x - r_2 - r_1}{x - r_2 + r_1}\right) = (\bar{x}, \bar{y}).$$

The birational map ρ^{-1} from E_e to E is given by $\rho^{-1}(\bar{x}, \bar{y}) = (\frac{(r_2-r_1)\bar{y}-(r_1+r_2)}{\bar{y}-1}, \frac{-2r_1t_1(\bar{y}+1)}{\bar{x}(\bar{y}-1)})$.

We may now define the homomorphism $\psi_e = \rho\psi\rho^{-1}$. We will compute an explicit form for ψ_e. Let $a = r_1 + r_2$ and $b = r_2 - r_1$. Then

$$\begin{aligned}
\psi_e(\bar{x}, \bar{y}) &= \rho\psi\rho^{-1}(\bar{x}, \bar{y}) \\
&= \rho\left(\frac{c_1(b\bar{y} - a)^p}{(\bar{y} - 1)^p}, \frac{c_2(-2r_1t_1(\bar{y} + 1))^p}{(\bar{x}(\bar{y} - 1))^p}\right) \\
&= \left(\frac{2t_1(c_1(\frac{b\bar{y}-a}{\bar{y}-1})^p - r_2)}{c_2(\frac{-2r_1t_1(\bar{y}+1)}{\bar{x}(\bar{y}-1)})^p}, \frac{c_1(\frac{b\bar{y}-a}{\bar{y}-1})^p - a}{c_1(\frac{b\bar{y}-a}{\bar{y}-1})^p - b}\right) \\
&= \left(\frac{\bar{x}^p(m_1 - m_2\bar{y}^p)}{m_3(\bar{y}^p + 1)}, \frac{m_4\bar{y}^p - m_5}{m_6\bar{y}^p - m_7}\right)
\end{aligned}$$

where $m_1 = c_1a^p - r_2$, $m_2 = c_1b^p - r_2$, $m_3 = c_2r_1^pt_1^{p-1}$, $m_4 = c_1b^p - a$, $m_5 = c_1a^p - a$, $m_6 = c_1b^p - b$, and $m_7 = c_1a^p - b$ are constants in $\mathbb{F}_{p^2}$ which may be precomputed.

It follows that ψ_e can be computed naively using two Frobenius computations, 5 multiplications and two inversions. We can also use Montgomery's trick to replace two inversions with one inversion and one multiplication. So the homomorphism ψ_e is certainly efficiently computable.

6 Multiexponentiation

The point of the GLV method is to replace a large point multiplication $[n]P$ by a multiexponentiation $[n_0]P + [n_1]\psi(P)$. It is well known that the 2-dimensional case there is a joint sparse form (JSF) for double exponentiation which has a joint Hamming weight approximately 1/2 the bitlength of the

pair of exponents (see [26]). Write $l(n) = \lceil \log_2(n) \rceil$. If the point multiplication $[n]P$ is performed using a non-adjacent form representation of n then it requires $l(n)$ doubles and $l(n)/2$ additions. Similarly, the double exponentiation using the Solinas JSF requires $l(n)/2$ doubles and $l(n)/4$ additions. This makes the double exponentiation roughly twice the speed (ignoring precomputation) of a single exponentiation using simple signed expansions – but not if using windowing methods.

When considering m-dimensional versions of the GLV method one must consider higher-dimensional variants of the joint sparse form. This is even more serious when performing signature verification, which is already a multiexponentiation, combined with the GLV method (i.e., a $2m$-dimensional multiexponentiation). Grabner, Heuberger and Prodinger [15] and Proos [21] have considered this problem and given algorithms to construct such representations for any m. However, the performance improvement becomes small as m grows. Roughly, one expects to get a joint signed representation of the m exponents n_i such that each block of $m+1$ bits has two 'zero columns'. In other words, using a generalised JSF one expects the multiexponentiation to cost $1+l(n)/m$ doubles plus $(m-1)/(m+1)(1+l(n)/m)$ adds, compared with $l(n)/m$ doubles plus $log_2(n)/m$ adds for the naive method. This is still an improvement, but it is clear that taking an m-dimensional version is not in general m times faster than a 1-dimensional version.

Also note that the precomputation requirements grow significantly as m increases. This may or may not be a concern depending on the application. One way to avoid such problems is to not precompute and store all intermediate values of the windows, and to compute some values on demand.

In general, the choice of m depends on the platform and application.

7 Point multiplication in Montgomery form

Bernstein [4] and Brown [9] have discussed the analogue of multiexponentiation in the case when only x-coordinates are used (for example, using Montgomery elliptic curves). Using their results one can implement the GLV method in this manner.

Unlike the standard case things are not so simple and y-coordinates are needed to compute the initial steps. Hence, one cannot avoid "point decompression". In future work we will compare the cost of the GLV method when using Montgomery form with our results already obtained.

8 Hyperelliptic curves

Afficionados will have noticed that Theorem 1 holds (with minor modifications to the second part of property (2)) for arbitrary abelian varieties. We now present an analogue of Corollary 1 for hyperelliptic curves.

Let $C : y^2 = x^{2g+1} + f_{2g}x^{2g} + \cdots + f_1 x + f_0$ be a genus g curve over $\mathbb{F}_q$ with a single point at infinity. Consider the Jacobian of C over $\mathbb{F}_{q^m}$ and take a quadratic twist $C' : y^2 = x^{2g+1} + u f_{2g}x^{2g} + \cdots + u^{2g}x + u^{2g+1}f_0$ where $u \in \mathbb{F}_{q^m}$ is a non-square. The isomorphism $\psi : C \to C'$ is given by

$$\phi(x, y) = (ux, \sqrt{u}^{2g+1} y)$$

This map induces an isomorphism $\phi : \mathrm{Jac}(C) \to \mathrm{Jac}(C')$ over $\mathbb{F}_{q^{2m}}$ which can be explicitly calculated on the Mumford representation.

Our construction leads to the homomorphism $\psi = \phi\pi\phi^{-1}$ satisfying $\psi^m(D) + D = 0$ for $D \in \mathrm{Jac}(C')(\mathbb{F}_{q^m})$ and therefore, when m is a power of 2, one obtains an m-dimensional GLV method.

In this case, the speedup for key generation is crucial: counting the number of points on random Jacobians of cryptographic size in large characteristic is currently impractical, however our new approach is certainly feasible in practice.

On the other hand, Weil descent attacks are much more successful in higher genus. Indeed, as discussed in Section 10, even the case $m = 2$ is potentially vulnerable to Weil descent attacks. Hence one needs to increase the size of q to attain the required security level and so the benefit of our ideas in this setting is unclear.

9 Experimental results

We now give some timing comparisons for the computation of $[n]P$. We consider the case where the point P is not fixed, and so include the cost of all precomputations needed for window methods and multiexponentiation.

Consider an elliptic curve implementation which is to provide security at the standard AES-128 bit level. In this case, from an implementation point of view, it is hard to resist the obvious attractions of the Mersenne prime $p = 2^{127} - 1$, which is also used in Bernstein's `surface1271` genus 2 implementation [5]. This prime supports a very fast modular reduction algorithm. Note that since $p \neq 5 \bmod 8$ the previously described key generation process is not applicable here. However it can easily be modified to handle this case as well, although the homomorphism becomes a little more complex.

Over this prime field we will use the elliptic curve

$$y^2 = x^3 - 3x + 44$$

defined over the field $\mathbb{F}_p$, whose quadratic twist over $\mathbb{F}_{p^2}$ has $\#E'(\mathbb{F}_{p^2}) = p + 1 + t^2 - 2p$ points on it, where

$$\#E'(\mathbb{F}_{p^2}) = \texttt{3FFFFFFFFFFFFFFFFFFFFFFFFFFFFFE09C5F010948D9D930E79156D8BA3CAF5}$$

is a prime. The curve was quickly found using a modified version of Schoof's algorithm.

Since $p \equiv 7 \bmod 8$, we can choose to represent values in $\mathbb{F}_{p^2}$ as $a+ib$, or $[a, b]$, where $i = \sqrt{-1}$. So now choose $u = 2 + i$ which is a non-square in $\mathbb{F}_{p^2}$. Note also that $p \equiv 2 \bmod 5$. Our homomorphism is this case simplifies to

$$\psi(x, y) = (\omega_x.\bar{x}, \omega_y.\bar{y})$$

where $\bar{x}$ denotes the conjugate of x, and

$$\omega_x = [(p + 3)/5, (3p + 4)/5]$$
$$\omega_y = [\texttt{12B04E814703D49C1AFAC10F88821962}, \texttt{426B94A2AD451F296F755142FE73FB62}]$$

For comparison purposes we use an elliptic curve $E(\mathbb{F}_p)$ defined with respect to a 256-bit pseudo-Mersenne modulus p, which provides the same level of security, and we will compare times for a full point multiplication. Our implementation will use standard Jacobian coordinates, as specified in the IEEE-1363 standard. Whereas improved parameterisations and formulae have since become available, we point out that these would benefit both implementations more or less equally. (Having said that, when the point doubling and addition formulae are implemented over $\mathbb{F}_{p^2}$ we note that field squarings are now fully 50% faster than field multiplications, and that field inversions are not

quite as bad as they would be over $\mathbb{F}_p$, for a double-length p, and hence implementations based on affine coordinates may be of interest in very space constrained environments.)

First, a short analysis of the speed-up to be expected using the new idea. When implementing $\mathbb{F}_{p^2}$ arithmetic, extension field multiplication using Karatsuba costs $3m$, where m is the cost of an $\mathbb{F}_p$ multiplication. Also, using an obvious trick, a field squaring costs $2m$. Since $\mathbb{F}_p$ squarings do not arise in these calculations, we ignore them. When implementing the double-multiplication algorithm using a Joint Sparse Form [26, 16], the cost of the calculation for each bit of the multiplier is 1 point doubling, plus 0.5 "mixed" point additions. Substituting the costs of point doubling and mixed point addition using standard Jacobian coordinates and IEEE-1363 formulae, gives a total cost of $(3(3m) + 6(2m) + (8(3m) + 3(2m))/2 = 36m$ per bit. For the 128 bit size of the multipliers required for the GLV method, this amounts to $128.36m = 4608m$ for the whole calculation.

For the 256-bit curve, the cost of a point multiplication depends on the windowing method used. Here we use a standard wNAF method, with a sliding window size of 4. Since we only perform additions at most one quarter of the time the expected cost will then be approximately $(4M+4S)+0.25(8M+3S)$ per bit where M is the cost of an $\mathbb{F}_p$ multiplication and S the cost of an $\mathbb{F}_p$ squaring. For simplicity we will assume that $M = S$, and we ignore the cost of the precomputation, and so the total cost for a 256-bit point multiplication works out as roughly $2752M$.

It is useful to also compare with the operation counts that arise when implementing straightforward point multiplication on $E(\mathbb{F}_{p^2})$ *without* using the new homomorphism, so we include these as well.

In our implementations we averaged the cost over 10^5 point multiplications, and the results are presented in Table 1. These results validate our rough analysis above. However we also include in the table the often neglected costs of field additions and subtractions. Note that when implementing $\mathbb{F}_{p^2}$ arithmetic, each multiplication using Karatsuba also requires five $\mathbb{F}_p$ additions/subtractions, so the number of these operations increases dramatically.

Table 1. Point multiplication operation counts

	$\mathbb{F}_p$ muls	$\mathbb{F}_p$ adds/subs
$E(\mathbb{F}_p)$ (256-bit p)	2729	3824
$E(\mathbb{F}_{p^2})$ using GLV (128-bit p)	4499	12061
$E(\mathbb{F}_{p^2})$ without GLV (128-bit p)	6596	19023

Clearly the superiority (or otherwise) of the new method depends on the efficiency with which 128-bit and 256-bit field multiplications and additions/subtractions can be implemented on particular platforms. To this end we have initially implemented both methods on two widely differing platforms, a 1.666GHz 64-bit Intel Core 2, and on an 8-bit 4MHz Atmel Atmega128L chip (which is a popular choice for Wireless Sensor Network nodes).

9.1 8-bit processor implementation

On the small Atmega128L 8-bit processor, the multiplication times will dominate, so this function was written in optimal loop-unrolled assembly language. We use the MIRACL library [22], which includes tools for the automatic generation of such code (and which holds the current speed record

in its class [23]). An 8-bit processor is generally happier with a smaller base field size, so it is to be expected that simply moving from a 256-bit field size to a 128-bit field size will bring some benefits. Up at the 256-bit field size, standard "school-boy" methods of multiprecision multiplication start to become slow, and should ideally begin to give way to Karatsuba-based methods, but in fact we are not in a comfort zone for either method – both are rather inefficient. Our tests indicate that for a 256-bit field $O(n^2)$ "school-boy" methods are still (just) a bit faster.

Table 2. Point multiplication timings – 8-bit processor

Atmel Atmega128L processor	Time (s)
$E(\mathbb{F}_p)$ (256-bit p)	7.45
$E(\mathbb{F}_{p^2})$ using GLV (128-bit p)	4.34
$E(\mathbb{F}_{p^2})$ without GLV (128-bit p)	6.61

As can be seem from these timings, our new method for ECC point multiplication is nearly 72% faster. However observe that there are two effects at work here – and that some of this improvement can be put down to the fact that a smaller word length processor prefers a smaller field size. Nonetheless the major speed-up is due to the applicability of the GLV method.

9.2 64-bit processor implementation

It has been eloquently observed by Avanzi [1], that software implementations over smaller prime fields (as required by our proposed method), where field elements can be stored in just a few CPU registers, suffer disproportionally when implemented using general purpose multi-precision libraries. This "Avanzi effect" would work against us here, as we are using the general purpose MIRACL library [22]. Special purpose libraries like the mpFq library [14] which generate field-specific code, and implementations which work hard to squeeze out overheads, such as Bernstein's implementations [5] are always going to be faster.

In the context of a 64-bit processor, while one might hope that timings would be dominated by the $O(n^2)$ base field multiplication operations, for small values of n the $O(n)$ contribution of the numerous base field additions and subtractions becomes significant, as also observed by Gaudry and Thomé [14]. Observe that on the 64-bit processor a 128-bit field element requires just $n = 2$ (and indeed the description as "multi-precision" should really give way to "double precision"). Therefore it is to be expected that the speed-up we can achieve in this case will be less than might have been hoped.

So is our new method faster? There is really only one satisfactory way to resolve the issue – and that is to identify the fastest known $E(\mathbb{F}_p)$ implementation on a 64-bit processor, and try to improve on it. We understand that the current record is that announced by Gaudry and Thomé at SPEED 2007 [14], using an implementation of Bernstein's `curve25519` [3]. This record is in the setting of an implementation of the elliptic curve Diffie-Hellman method, which requires a single point multiplication to determine the shared secret key.

We point out that the implementation and optimizations of `curve25519` are very much in the sole context of an efficient Diffie-Hellman implementation – ours is general purpose and immediately applicable to a wide range of ECC protocols. In particular the implementation of `curve25519` uses

Montgomery's parameterisation of an elliptic curve, and is not required to maintain a y point coordinate, and hence can achieve compression of the public key without the calculation of a field square root.

On the other hand we have the use of a particularly nice modulus $2^{127}-1$, which brings many benefits. For example a base field square root of a quadratic residue x can be calculated as simply $x^{2^{125}}$.

In order to be competitive we wrote a specialised hand-crafted x86-64 assembly language module to handle the base field arithmetic, and integrated this with the MIRACL library. Given that each field element can be stored in just two 64-bit registers, this code is quite short, and did not take long to generate, optimize and test.

To obtain a fair comparison, we utilise the very useful and easy to use `eBats` facility [8]. This measures the runtime in terms of CPU clock cycles. Our cycle counts are for an Intel Core 2 processor. Note that for technical reasons there is a small natural variance in the number of clock cycles recorded, so these are averaged figures.

Table 3. Point multiplication timings – 64-bit processor

Intel Core 2 processor	Clock cycles
$E(\mathbb{F}_p)$ (256-bit p)	386,000
$E(\mathbb{F}_{p^2})$ (128-bit p) without compression	359,344
$E(\mathbb{F}_{p^2})$ (128-bit p) with compression	375,736

Our eBat can be downloaded from `ftp://ftp.computing.dcu.ie/pub/crypto/gls1271-2.tar`. Profiling the code reveals that our with-compression version spends 48.32% of its time doing base field multiplications and squarings, and 19.18% of the time doing base field additions and subtractions. Calculating and managing the joint sparse form takes 4.84% of the total time, and approximately 6% of the time is required for modular inversion.

10 Security implications

Our homomorphisms (at least, in the case when ϕ is an isomorphism) define equivalence classes of points in $E'(\mathbb{F}_{p^m})$ of size $2m$ by $[P] = \{\pm\psi^i(P) : 0 \leq i < m\}$. By the methods of Gallant-Lambert-Vanstone [11] and Wiener-Zuccherato [27] one can perform the Pollard algorithms for the discrete logarithm problem on these equivalence classes. This speeds up the solution of the discrete logarithm problem by a factor of $\sqrt{m}$ over previous techniques.

A more serious threat to our proposal comes from the Weil descent philosophy, and in particular the work of Gaudry [13]. Gaudry gives an algorithm for the discrete logarithm problem in $E'(\mathbb{F}_{p^n})$ requiring time $O(p^{2-4/(2n+1)})$ (with bad constants) which, in principle, beats the Pollard methods for $n \geq 3$. Gaudry's method also applies to abelian varieties: if A is an abelian varitey of dimension d over $\mathbb{F}_{p^n}$ then the algorithm has complexity $O(p^{2-4/(2dn+1)})$. Hence, for Jacobians of genus 2 curves over $\mathbb{F}_{p^2}$ one has an algorithm running in time $O(p^{1.55})$, rather than the Pollard complexity of $O(p^2)$.

Gaudry's method is still exponential time and so one can secure against it by increasing the parameters. For example, to achieve 128-bit security level using our construction with genus 2 curves

over $\mathbb{F}_{p^2}$ (or elliptic curves over $\mathbb{F}_{p^4}$) one should take p to be approximately 80 bits rather than the desired 64 bits.

Acknowledgements

We thank Dan Bernstein, Billy Brumley, Jinhui Chao, Pierrick Gaudry, Alfred Menezes, Yasuyuki Nogami and Fre Vercauteren for suggestions and comments.

References

1. R. Avanzi, Aspects of Hyperelliptic Curves over Large Prime Fields in Software Implementations, CHES 2004, Springer LNCS 3156 (2004), 148–162
2. R. Avanzi, H. Cohen, C. Doche, G. Frey, T. Lange, K. Nguyen and F. Vercauteren, Handbook of elliptic and hyperelliptic cryptography, Chapman and Hall/CRC, 2006.
3. D. J. Bernstein, Curve25519: New Diffie-Hellman Speed Records, in M. Yung et al (eds), PKC 2006, Springer LNCS 3958 (2006) 207–228.
4. D. J. Bernstein, Differential addition chains, preprint.
5. D. J. Bernstein, Elliptic vs. hyperelliptic, part 1 ECC 2006, Toronto, Canada http://www.cacr.math.uwaterloo.ca/conferences/2006/ecc2006/slides.html
6. D. J. Bernstein and T. Lange, Faster addition and doubling on elliptic curves, in K. Kurosawa (ed), Asiacrypt 2007, Springer LNCS 4833 (2007) 29–50.
7. D. J. Bernstein and T. Lange, Inverted Edwards coordinates, in S. Boztas and H.-F. Lu (eds.), AAECC 2007, Springer LNCS 4851 (2007) 20–27.
8. eBATS: ECRYPT Benchmarking of Asymmetric Systems, http://www.ecrypt.eu.org/ebats/
9. D. R. L. Brown, Multi-Dimensional Montgomery Ladders for Elliptic Curves, eprint 2006/220.
10. S. D. Galbraith and M. Scott, Exponentiation in pairing-friendly groups using homomorphisms, preprint 2008. http://eprint.iacr.org/2008/117
11. R. P. Gallant, R. J. Lambert and S. A. Vanstone, Improving the parallelized Pollard lambda search on anomalous binary curves, *Math. Comp.*, **69** (2000), 1699-1705.
12. R. P. Gallant, R. J. Lambert and S. A. Vanstone, Faster Point Multiplication on Elliptic Curves with Efficient Endomorphisms. In J. Kilian (Ed.), CRYPTO 2001, Springer LNCS 2139 (2001), 190–200.
13. P. Gaudry, Index calculus for abelian varieties and the elliptic curve discrete logarithm problem, to appear in *J. Symbolic Comput.*
14. P. Gaudry and E. Thome, The mpFq library and implementing curve-based key exchanges. SPEED workshop presentation, Amsterdam, June 2007
15. P. J. Gabner,C. Heuberger and H. Prodinger, Distribution results for low-weight binary representations for pairs of integers, *Theoretical Comp. Sci.*, **319** (2004) 307–331.
16. D. Hankerson, A. J. Menezes and S. Vanstone, Guide to elliptic curve cryptography, Springer (2004).
17. T. Iijima, K. Matsuo, J. Chao and S. Tsujii, Costruction of Frobenius maps of twists elliptic curves and its application to elliptic scalar multiplication, SCIS 2002.
18. A. J. Menezes, Another look at HMQV, *J. Mathematical Cryptology*, 1 (2007), 47-64.
19. B. Möller, Algorithms for multi-exponentiation. In S. Vaudenay and A. M. Youssef (Eds.), SAC 2001, Springer LNCS 2259 (2001), 165–180.
20. Y. Nogami and Y. Morikawa, Fast generation of elliptic curves with prime order over $\mathbb{F}_{p^{2^c}}$, in Proceedings International Symposium on Information Theory 2003.
21. J. Proos, Joint sparse forms and generating zero solumns when combing, Technical report CORR 2003-23, CACR (2003)
22. M.Scott, MIRACL – Multiprecision Integer and Rational Arithmetic C/C++ Library, http://ftp.computing.dcu.ie/pub/crypto/miracl.zip, 2008

23. M. Scott and P. Szczechowiak, Optimizing Multiprecision Multiplication for Public Key Cryptography, preprint 2007. http://eprint.iacr.org/2007/299
24. F. Sica, M. Ciet, J.-J. Quisquater, Analysis of the Gallant-Lambert-Vanstone method based on efficient endomorphisms: Elliptic and hyperelliptic curves, in K. Nyberg and H. M. Heys (eds.), SAC 2002, Springer LNCS 2595 (2003) 21–36.
25. J. H. Silverman. *The Arithmetic of Elliptic Curves*. Graduate Texts in Mathematics 106. Springer-Verlag, 1986.
26. J. A. Solinas, Low-weight binary representations for pairs of integers, Technical report CORR 2001–41, CACR, 2001.
27. M. J. Wiener and R. J. Zuccherato, Faster Attacks on Elliptic Curve Cryptosystems. In S. Tavares and H. Meijer (Eds.), SAC 1998, Springer LNCS 1556 (1999), 190–200.