Analysis and optimization of elliptic-curve single-scalar multiplication

Daniel J. Bernstein and Tanja Lange

Dedicated to Henri Cohen on the occasion of his sixtieth birthday.

Abstract. Let P be a point on an elliptic curve over a finite field of large characteristic. Exactly how many points $2P, 3P, 5P, 7P, 9P, \ldots, mP$ should be precomputed in a sliding-window computation of nP? Should some or all of the points be converted to affine form, and at which moments during the precomputation should these conversions take place? Exactly how many field multiplications are required for the resulting computation of nP? The answers depend on the size of n, the $\mathbf{I}/\mathbf{M}$ ratio, the choice of curve shape, the choice of coordinate system, and the choice of addition formulas. This paper presents answers that, compared to previous analyses, are more carefully optimized and cover a much wider range of situations.

1. Introduction

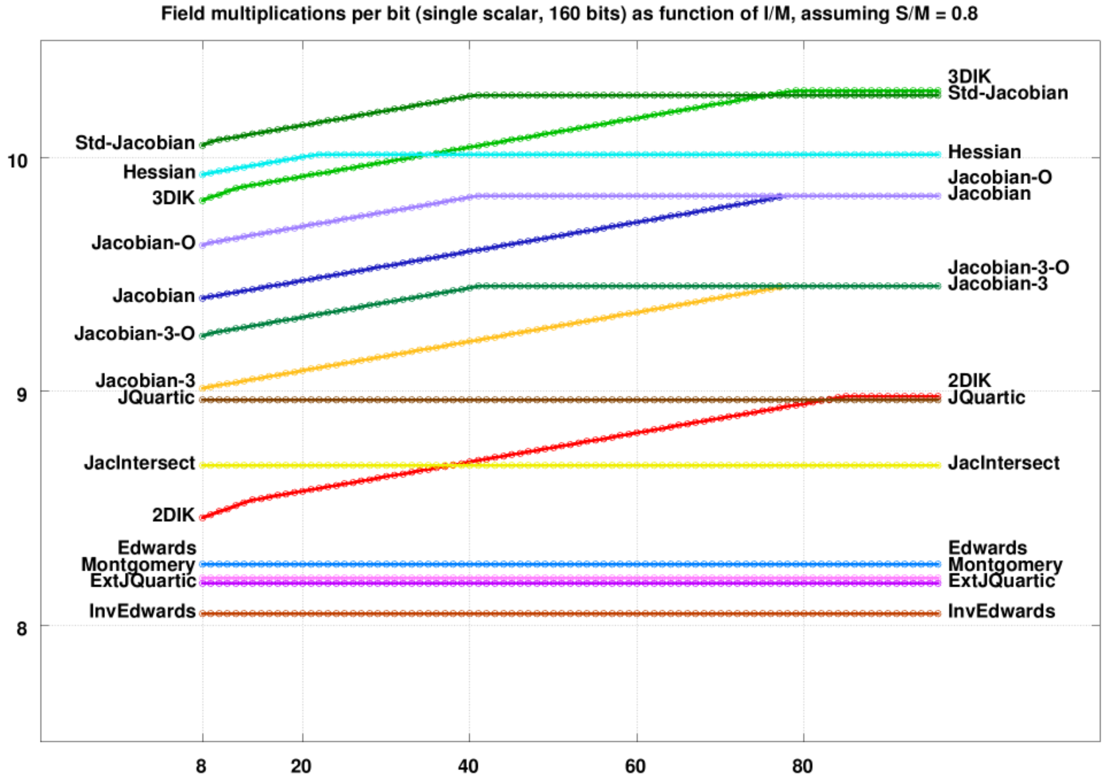
Consider the problem of computing a scalar multiple nP of a point P on an elliptic curve over a finite field of large characteristic. Cohen, Miyaji, and Ono, in a classic paper [**12**], analyzed the cost of a wide variety of scalar-multiplication methods and issued concrete recommendations for the lowest-cost methods. For example, for 160-bit scalars, Cohen, Miyaji, and Ono recommended one method using $4\mathbf{I} + 1488.4\mathbf{M}$ (i.e., 4 field inversions and on average 1488.4 field multiplications; a field squaring is implicitly counted as 0.8 field multiplications) and another method using $1610.2\mathbf{M}$. Both methods produce nP in Jacobian coordinates; the second method is better if $\mathbf{I}/\mathbf{M}$ is large.

In this paper we identify faster scalar-multiplication methods. For example, for 160-bit scalars, we obtain nP in Jacobian coordinates using just $1\mathbf{I} + 1495.8\mathbf{M}$ when $\mathbf{I}/\mathbf{M}$ is small, or $1573.8\mathbf{M}$ when $\mathbf{I}/\mathbf{M}$ is large. Even better, for curves that allow the "$a_4 = -3$" speedup, we obtain nP in Jacobian coordinates using just

2000 *Mathematics Subject Classification.* Primary 94A60. Secondary 12Y05, 14Q05.

Permanent ID of this document: `8ac889630fe4d44913b92cc5914aa01b`. Date: 2007.12.04. This work was supported in part by the National Science Foundation under grant ITR–0716498 and in part by the European Commission through the IST Programme under Contract IST–2002–507932 ECRYPT. Parts of this work were carried out while the first author was visiting Technische Universiteit Eindhoven.

FIGURE 1.1. Multiplications per bit for 160-bit scalars, as function of the $\mathbf{I}/\mathbf{M}$ ratio, assuming $\mathbf{S}/\mathbf{M} = 0.8$.

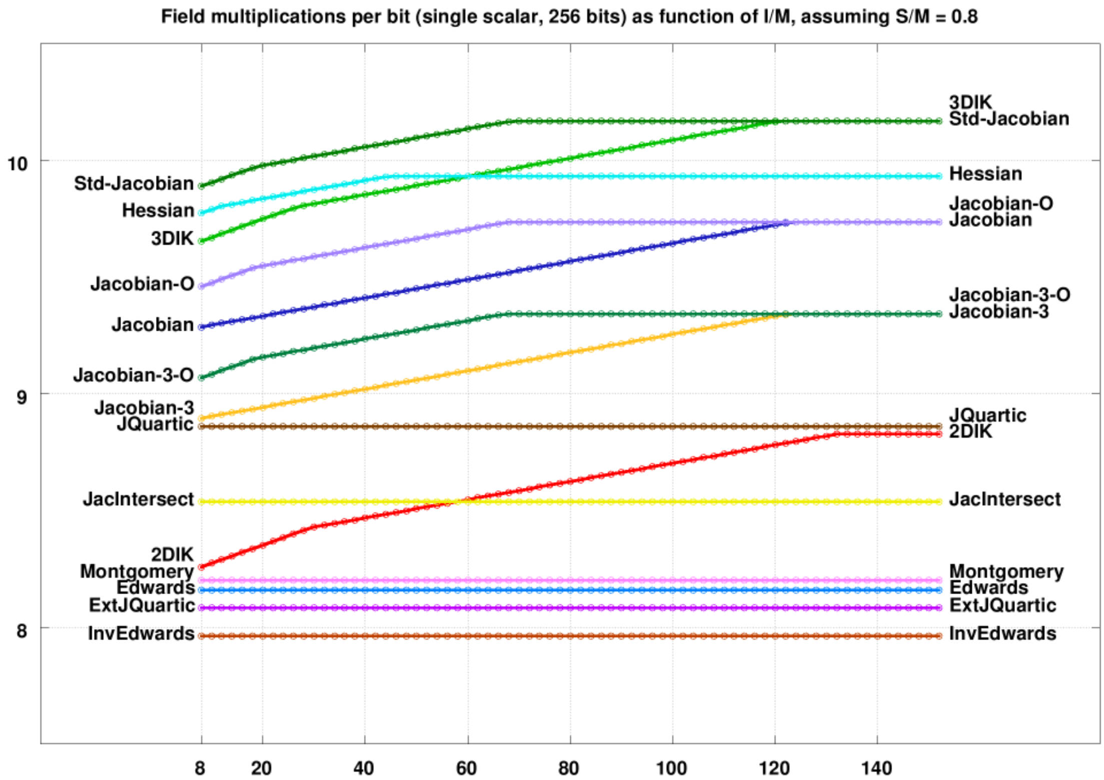


$1\mathbf{I} + 1434.1\mathbf{M}$ or $1511.9\mathbf{M}$. Even better, for curves that can be transformed to Edwards form, we obtain nP in inverted Edwards coordinates using just $1287.8\mathbf{M}$.

There are several reasons that, compared to the analysis in [**12**], we find lower costs for scalar multiplication:

- We use faster formulas for elliptic-curve addition and doubling. For example, Cohen, Miyaji, and Ono say that doubling in projective coordinates takes $7\mathbf{M} + 5\mathbf{S} = 11\mathbf{M}$; we replaced a multiplication with a squaring, reducing the cost to $6\mathbf{M} + 6\mathbf{S} = 10.8\mathbf{M}$.
- We assume that curves are sensibly chosen with small parameters so that multiplications by those parameters have negligible cost. For example, choosing a small curve parameter "a_4" in projective coordinates reduces the cost of doubling to $5\mathbf{M} + 6\mathbf{S} = 9.8\mathbf{M}$.
- We use "fractional windows": precomputing $2P, 3P, 5P, 7P, \ldots, mP$ for any odd $m \geq 3$. See Section 3. For example, for 160-bit scalars in Jacobian coordinates without inversions, we recommend precomputing $2P, 3P, 5P, 7P, 9P, 11P, 13P$. Cohen, Miyaji, and Ono impose the common restriction $m \in \{3, 7, 15, 31, \ldots\}$ and are thus forced to precompute $2P, 3P, 5P, 7P, 9P, 11P, 13P, 15P$. The benefit of fractional windows depends on how far the optimal m is from a power of 2.
- We further optimize the precomputation by allowing 0 inversions, 1 inversion, 2 inversions, or 3 inversions at carefully selected moments. See

FIGURE 1.2. Multiplications per bit for 256-bit scalars, as function of the $\mathbf{I}/\mathbf{M}$ ratio, assuming $\mathbf{S}/\mathbf{M} = 0.8$.



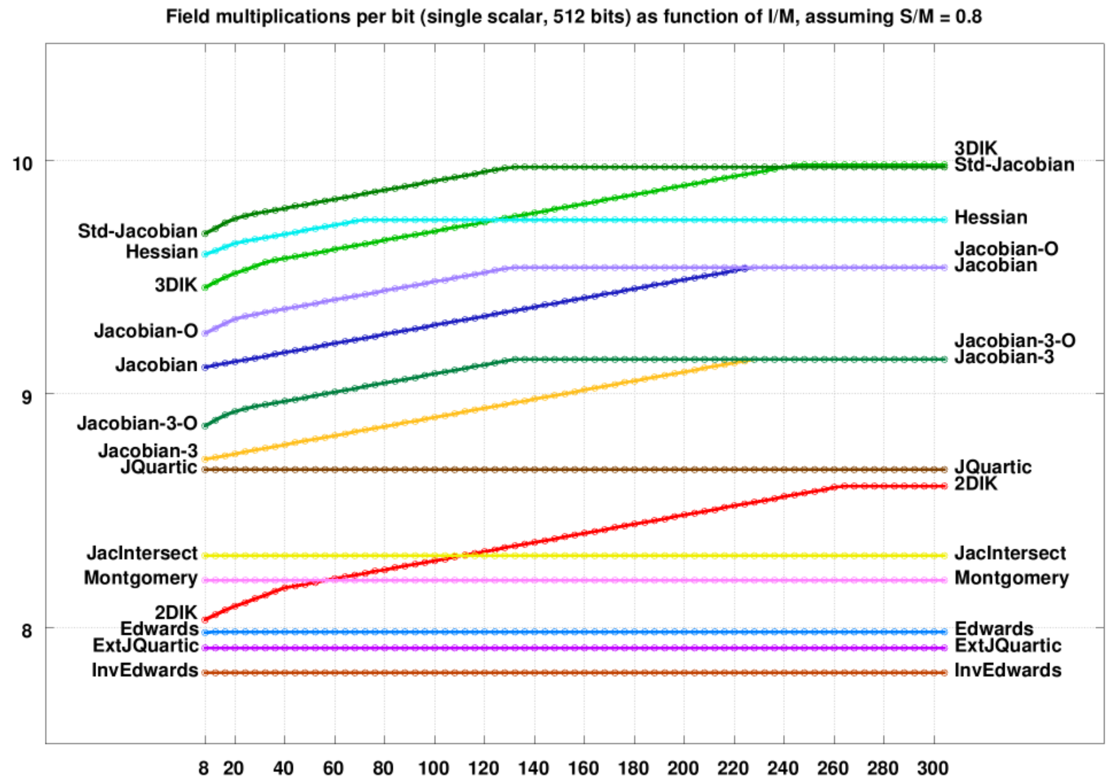
Section 4. Cohen, Miyaji, and Ono consider only two possibilities, namely 0 inversions and $\lg(m+1)$ inversions.

- We incorporate a faster use of 1 inversion proposed by Dahmen, Okeya, and Schepers in the recent paper [**13**]. Our graphs also include "Jacobian-O" and "Jacobian-3-O" without this speedup to show how much the speedup helps.
- We consider parameter families that allow further speedups in additions and doublings: "Projective-3" and "Jacobian-3" and "2DIK" and "3DIK." See Section 2. For example, choosing $a_4 = -3$ in projective coordinates reduces the cost of doubling to $7\mathbf{M} + 3\mathbf{S} = 9.4\mathbf{M}$.
- We take account of different curve shapes allowing faster additions and doublings: "Hessian" and "JacIntersect" and "JQuartic" and "ExtJQuartic" and "Edwards" and "InvEdwards." See Section 2.

Some papers in the last ten years have updated parts of the analysis of [**12**] — for example, Dahmen, Okeya, and Schepers included a comparison of their 1-inversion method to 0 inversions and $\lg(m+1)$ inversions for Std-Jacobian — but our analysis is much more comprehensive than any previous analysis in the literature.

Figure 1.1 presents our results for 160-bit scalars in graphical form. For example, the graph includes a small circle (red in color displays) next to "2DIK" at horizontal position 8 and vertical position ≈ 8.4. This small circle indicates that 160-bit elliptic-curve scalar multiplication in doubling-oriented Doche/Icart/Kohel coordinates, with the best parameters that we found, uses ≈ 8.4 field multiplications

FIGURE 1.3. Multiplications per bit for 512-bit scalars, as function of the $\mathbf{I}/\mathbf{M}$ ratio, assuming $\mathbf{S}/\mathbf{M} = 0.8$.



per bit if $\mathbf{I}/\mathbf{M} = 8$. Here $\mathbf{I}/\mathbf{M}$ is the ratio between the time for a field inversion and the time for a field multiplication. Red circles at subsequent horizontal positions show how scalar multiplication slows down as $\mathbf{I}/\mathbf{M}$ increases; the maximum, just below 9 field multiplications per bit (with no inversions), is reached once $\mathbf{I}/\mathbf{M}$ increases to about 85. Other colors show similar results for other coordinate systems.

Table 4.1 presents the same results in tabular form, with additional details. For example, the table row that begins "160 2DIK 2**I**" shows the best parameters that we found using 2 inversions for 160-bit scalar multiplication in doubling-oriented Doche/Icart/Kohel coordinates. The "m" column in the same row lists "9" to indicate that we precomputed $2P, 3P, 5P, 7P, 9P$; here we used one inversion to convert $2P$ to affine coordinates, and the other inversion to convert $3P, 5P, 7P, 9P$ to affine coordinates. The "Multiplications and squarings" column lists the cost of scalar multiplication with these parameters, namely "$598.9\mathbf{M}+923.4\mathbf{S} \approx 1337.6\mathbf{M}$" plus (implicitly) the aforementioned two inversions; this means about $1353.6\mathbf{M}$ for $\mathbf{I}/\mathbf{M} = 8$, i.e., about 8.46 multiplications per bit. The "$\mathbf{I}/\mathbf{M}$" column lists "≤ 14.0" to indicate that 2 inversions are preferable to 1 inversion when $\mathbf{I} \leq 14.0\mathbf{M}$. For details on the precomputations and in particular on how the inversions are used we refer to Section 4.

Operation counts for projective coordinates appear in Table 4.1 but are above the top of Figure 1.1. We could have changed the scale of Figure 1.1 to include projective coordinates, but the other coordinate systems would then have been uncomfortably squished.

To emphasize the importance of speedups in elliptic-curve addition, we analyzed not only Jacobian and Jacobian-3 using the best speeds known, but also "Std-Jacobian" using the speeds most commonly quoted in the literature. The Std-Jacobian results in Figure 1.1 are considerably worse than the Jacobian and Jacobian-3 results. The graph also shows the striking advantage of recent advances in curve shapes, notably Edwards and ExtJQuartic and InvEdwards. InvEdwards is the current speed leader.

Figure 1.2 and Table 4.2 present similar results for 256-bit scalars. Figure 1.3 and Table 4.3 present similar results for 512-bit scalars. The increase in bit size reduces the number of curve additions per bit, saving time per bit and generally reducing the vertical height of the graphs. Some systems benefit slightly more than others; for example, the increase in bit size noticeably reduces the gap between 2DIK and Edwards for small $\mathbf{I}/\mathbf{M}$.

We did not consider $\mathbf{I}/\mathbf{M}$ ratios below 8. As far as we can tell, the only implementations of large-characteristic field arithmetic with $\mathbf{I}/\mathbf{M} < 8$ are implementations in which multiplication is very poorly optimized.

Like Cohen, Miyaji, and Ono, we assume that $\mathbf{S}/\mathbf{M} = 0.8$ and that field additions, field subtractions, etc. take negligible time. These assumptions are standard but debatable. Our analysis can easily accommodate changes in these assumptions, in the same way that it accommodates variations in curve shapes, coordinate systems, addition formulas, etc.; our software automatically and efficiently identifies optimal parameters given the costs of elliptic-curve operations.

2. Fast addition on elliptic curves

Each elliptic curve over a field k of large characteristic can be written in Weierstrass form

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

for some $a_2, a_4, a_6 \in k$ with $4a_6a_2^3 - a_4^2a_2^2 - 18a_6a_4a_2 + 4a_4^3 + 27a_6^2 \neq 0$. The group of k-rational points is denoted by $E(k)$; it contains the affine points $(x_1, y_1) \in k \times k$ satisfying $y_1^2 = x_1^3 + a_2x_1^2 + a_4x_1 + a_6$ and one point at infinity.

The standard formulas to add $(x_1, y_1), (x_2, y_2)$ on E with $(x_2, y_2) \neq (x_1, -y_1)$ are given by $(x_1, y_1) + (x_2, y_2) = (x_3, y_3) = (\lambda^2 - a_2 - x_1 - x_2, \lambda(x_1 - x_3) - y_1)$ where

$$\lambda = \begin{cases} \dfrac{y_1 - y_2}{x_1 - x_2} & \text{for} \quad (x_1, y_1) \neq (x_2, y_2), \\ \dfrac{3x_1^2 + 2a_2x_1 + a_4}{2y_1} & \text{for} \quad (x_1, y_1) = (x_2, y_2). \end{cases}$$

This means that an addition takes $1\mathbf{I}+2\mathbf{M}+1\mathbf{S}$ while a doubling takes $1\mathbf{I}+2\mathbf{M}+2\mathbf{S}$ and one multiplication with the curve parameter a_2. It is easy to find an isomorphic curve with $a_2 = 0$; we assume $a_2 = 0$ for projective and Jacobian coordinates. As mentioned in Section 1, we assume that curves are sensibly chosen with small curve parameters; we omit the cost of multiplication by curve parameters, and we omit the cost of field additions and subtractions. See [**3**] for complete operation counts that include these costs.

In this section we present twelve different elliptic-curve coordinate systems that allow inversion-free addition and inversion-free doubling. We start with systems that are visibly related to curves in Weierstrass form in the sense that a point with $Z_1 = 1$ has (X_1, Y_1) satisfying the Weierstrass equation. These systems are projective coordinates, Jacobian coordinates, doubling-oriented Doche/Icart/Kohel curves, and tripling-oriented Doche/Icart/Kohel curves. We then present Hessian curves, Edwards curves, Jacobi-quartic curves, and curves given as Jacobi intersections; these forms start with different defining equations.

For each of these systems we give the curve equation, explain how points are represented, state the neutral element of the group law, state a negation formula, and present an explicit map to Weierstrass form. We also state, in Tables 2.1 and 2.2, the number of field inversions, field multiplications, and field squarings for each of these systems for each of the following operations:

- ADD is the cost of a general *addition*.
- reADD is the cost of a *readdition*, i.e., an addition in which one of the summands has been added before. A readdition saves time when it reuses cached results from the previous addition.
- mADD is the cost of a *mixed addition*, i.e., an addition in which Z_2 is known to be 1.
- mmADD is the cost of an addition in which both Z_1 and Z_2 are known to be 1.
- DBL is the cost of a *doubling*, i.e., an addition in which both inputs are known to be equal.
- mDBL is the cost of a doubling in which Z_1 is known to be 1.
- SCALE is the cost of *scaling* a point to obtain $Z_1 = 1$. This cost always includes $\mathbf{I}$, the cost of a field inversion.
- xSCALE is the cost of *scaling an extra point*, so simultaneously scaling m points uses 1 SCALE and $m-1$ xSCALE. “Montgomery’s trick” computes $1/Z_1$ and $1/Z_2$ as $Z_2(1/Z_1Z_2)$ and $Z_1(1/Z_1Z_2)$, using $3\mathbf{M}+\mathbf{I}$ rather than $2\mathbf{I}$, so xSCALE is $\mathbf{I} - 3\mathbf{M}$ smaller than SCALE.

Our “Explicit-Formulas Database” (EFD) [**3**] is a collection of explicit formulas for these operations, justifying the operation counts in Tables 2.1 and 2.2. (The EFD also contains formulas for triplings in many coordinate systems; see [**2**] for an analysis of the importance of triplings.) The EFD contains the best formulas we could find in the literature. It also contains many additional speedups that we found and that are published only in the EFD.

The EFD is updated regularly to include the latest and fastest formulas. This implies that the tables in this paper reflect our current knowledge but are subject to change. The strategies we used to generate the tables and graphs are completely modular and can be applied to modified counts. We plan to integrate this paper’s tables and graphs into the EFD so that they are updated automatically.

Projective coordinates. A point (x, y) on a Weierstrass-form elliptic curve $y^2 = x^3+a_4x+a_6$ is represented as $(X : Y : Z)$ satisfying $Y^2Z = X^3+a_4XZ^2+a_6Z^3$ and $(x, y) = (X/Z, Y/Z)$. Here $(X : Y : Z) = (\lambda X : \lambda Y : \lambda Z)$ for all nonzero λ. The negative of $(X : Y : Z)$ is $(X : -Y : Z)$. The neutral element is represented as $(0 : 1 : 0)$. The group operation is directly related to that in affine Weierstrass form.

Chudnovsky and Chudnovsky in [**10**, formulas (4.4i) and (4.4ii)] presented explicit formulas for group operations in projective coordinates. Those formulas are still state-of-the-art except for some $\mathbf{S}-\mathbf{M}$ tradeoffs, replacing multiplications with

TABLE 2.1. Cost of addition, readdition, etc. in various elliptic-curve coordinate systems

Curve shape	ADD	reADD	mADD	mmADD
2DIK	$12\mathbf{M}+5\mathbf{S}$	$12\mathbf{M}+5\mathbf{S}$	$8\mathbf{M}+4\mathbf{S}$	$4\mathbf{M}+4\mathbf{S}$
3DIK	$11\mathbf{M}+6\mathbf{S}$	$10\mathbf{M}+6\mathbf{S}$	$7\mathbf{M}+4\mathbf{S}$	$4\mathbf{M}+2\mathbf{S}$
Edwards	$10\mathbf{M}+1\mathbf{S}$	$10\mathbf{M}+1\mathbf{S}$	$9\mathbf{M}+1\mathbf{S}$	$6\mathbf{M}+1\mathbf{S}$
ExtJQuartic	$8\mathbf{M}+3\mathbf{S}$	$8\mathbf{M}+3\mathbf{S}$	$7\mathbf{M}+3\mathbf{S}$	$5\mathbf{M}+4\mathbf{S}$
Hessian	$12\mathbf{M}+0\mathbf{S}$	$12\mathbf{M}+0\mathbf{S}$	$10\mathbf{M}+0\mathbf{S}$	$8\mathbf{M}+0\mathbf{S}$
InvEdwards	$9\mathbf{M}+1\mathbf{S}$	$9\mathbf{M}+1\mathbf{S}$	$8\mathbf{M}+1\mathbf{S}$	$7\mathbf{M}+0\mathbf{S}$
JacIntersect	$13\mathbf{M}+2\mathbf{S}$	$11\mathbf{M}+2\mathbf{S}$	$11\mathbf{M}+2\mathbf{S}$	$8\mathbf{M}+2\mathbf{S}$
Jacobian	$11\mathbf{M}+5\mathbf{S}$	$10\mathbf{M}+4\mathbf{S}$	$7\mathbf{M}+4\mathbf{S}$	$4\mathbf{M}+2\mathbf{S}$
Jacobian-3	$11\mathbf{M}+5\mathbf{S}$	$10\mathbf{M}+4\mathbf{S}$	$7\mathbf{M}+4\mathbf{S}$	$4\mathbf{M}+2\mathbf{S}$
JQuartic	$10\mathbf{M}+3\mathbf{S}$	$9\mathbf{M}+3\mathbf{S}$	$8\mathbf{M}+3\mathbf{S}$	$5\mathbf{M}+2\mathbf{S}$
Projective	$12\mathbf{M}+2\mathbf{S}$	$12\mathbf{M}+2\mathbf{S}$	$9\mathbf{M}+2\mathbf{S}$	$5\mathbf{M}+2\mathbf{S}$
Projective-3	$12\mathbf{M}+2\mathbf{S}$	$12\mathbf{M}+2\mathbf{S}$	$9\mathbf{M}+2\mathbf{S}$	$5\mathbf{M}+2\mathbf{S}$
Std-Jacobian	$12\mathbf{M}+4\mathbf{S}$	$11\mathbf{M}+3\mathbf{S}$	$8\mathbf{M}+3\mathbf{S}$	$4\mathbf{M}+2\mathbf{S}$

TABLE 2.2. Cost of doubling and scaling in various elliptic-curve coordinate systems

Curve shape	DBL	mDBL	SCALE	xSCALE
2DIK	$2\mathbf{M}+5\mathbf{S}$	$1\mathbf{M}+5\mathbf{S}$	$1\mathbf{I}+2\mathbf{M}+1\mathbf{S}$	$5\mathbf{M}+1\mathbf{S}$
3DIK	$2\mathbf{M}+7\mathbf{S}$	$1\mathbf{M}+5\mathbf{S}$	$1\mathbf{I}+3\mathbf{M}+1\mathbf{S}$	$6\mathbf{M}+1\mathbf{S}$
Edwards	$3\mathbf{M}+4\mathbf{S}$	$3\mathbf{M}+3\mathbf{S}$	$1\mathbf{I}+2\mathbf{M}+0\mathbf{S}$	$5\mathbf{M}+0\mathbf{S}$
ExtJQuartic	$3\mathbf{M}+4\mathbf{S}$	$1\mathbf{M}+6\mathbf{S}$	$1\mathbf{I}+2\mathbf{M}+1\mathbf{S}$	$5\mathbf{M}+1\mathbf{S}$
Hessian	$7\mathbf{M}+1\mathbf{S}$	$3\mathbf{M}+3\mathbf{S}$	$1\mathbf{I}+2\mathbf{M}+0\mathbf{S}$	$5\mathbf{M}+0\mathbf{S}$
InvEdwards	$3\mathbf{M}+4\mathbf{S}$	$3\mathbf{M}+3\mathbf{S}$	$1\mathbf{I}+2\mathbf{M}+0\mathbf{S}$	$5\mathbf{M}+0\mathbf{S}$
JacIntersect	$3\mathbf{M}+4\mathbf{S}$	$2\mathbf{M}+4\mathbf{S}$	$1\mathbf{I}+3\mathbf{M}+0\mathbf{S}$	$6\mathbf{M}+0\mathbf{S}$
Jacobian	$1\mathbf{M}+8\mathbf{S}$	$1\mathbf{M}+5\mathbf{S}$	$1\mathbf{I}+3\mathbf{M}+1\mathbf{S}$	$6\mathbf{M}+1\mathbf{S}$
Jacobian-3	$3\mathbf{M}+5\mathbf{S}$	$1\mathbf{M}+5\mathbf{S}$	$1\mathbf{I}+3\mathbf{M}+1\mathbf{S}$	$6\mathbf{M}+1\mathbf{S}$
JQuartic	$2\mathbf{M}+6\mathbf{S}$	$1\mathbf{M}+4\mathbf{S}$	$1\mathbf{I}+2\mathbf{M}+1\mathbf{S}$	$5\mathbf{M}+1\mathbf{S}$
Projective	$5\mathbf{M}+6\mathbf{S}$	$3\mathbf{M}+5\mathbf{S}$	$1\mathbf{I}+2\mathbf{M}+0\mathbf{S}$	$5\mathbf{M}+0\mathbf{S}$
Projective-3	$7\mathbf{M}+3\mathbf{S}$	$3\mathbf{M}+5\mathbf{S}$	$1\mathbf{I}+2\mathbf{M}+0\mathbf{S}$	$5\mathbf{M}+0\mathbf{S}$
Std-Jacobian	$3\mathbf{M}+6\mathbf{S}$	$2\mathbf{M}+4\mathbf{S}$	$1\mathbf{I}+3\mathbf{M}+1\mathbf{S}$	$6\mathbf{M}+1\mathbf{S}$

squarings. We denote this system by "Projective" in our tables, and in principle also in our graphs, but the system is so slow that it is beyond the top of the graphs.

Doubling is faster if $a_4 = -3$. This choice includes about half of all isomorphism classes of elliptic curves over a finite field, and almost all isogeny classes; see [**9**]. We denote this system with $a_4 = -3$ by "Projective-3."

Jacobian coordinates. A point (x, y) on an elliptic curve $y^2 = x^3 + a_4x + a_6$ is represented as $(X : Y : Z)$ satisfying $Y^2 = X^3 + a_4XZ^4 + a_6Z^6$ and $(x, y) = (X/Z^2, Y/Z^3)$. Here $(X : Y : Z) = (\lambda^2X : \lambda^3Y : \lambda Z)$ for all nonzero λ. The negative of $(X : Y : Z)$ is $(X : -Y : Z)$. The neutral element is represented as $(1 : 1 : 0)$. The group operation is directly related to that in affine Weierstrass form.

Chudnovsky and Chudnovsky in [**10**, formulas (4.2ii) and (4.3i')] presented explicit formulas for group operations in Jacobian coordinates. Those formulas are still state-of-the-art except for some $\mathbf{S}-\mathbf{M}$ tradeoffs. In the tables and graphs we denote this system by "Jacobian."

As in projective coordinates, doubling is faster if $a_4 = -3$. This choice includes about half of all isomorphism classes of elliptic curves over a finite field. We denote this system with $a_4 = -3$ by "Jacobian-3."

To demonstrate the importance of optimized formulas, our graphs also include "Std-Jacobian," the operation counts for Jacobian coordinates appearing in [**12**, Section 2.3] and in many subsequent papers. We encourage implementors to copy the latest formulas from the EFD rather than using obsolete formulas.

Some papers describe readdition in Jacobian coordinates as "addition in Chudnovsky coordinates." The concept of readditions allows us to avoid treating "Chudnovsky coordinates" separately.

Doubling-oriented Doche/Icart/Kohel curves. A point (x, y) on an elliptic curve $y^2 = x^3 + ax^2 + 16ax$ is represented as $(X : Y : Z : Z^2)$ satisfying $Y^2 = ZX^3 + aZ^2X^2 + 16aZ^3X$ and $(x, y) = (X/Z, Y/Z^2)$. Here $(X : Y : Z : Z^2) = (\lambda X : \lambda^2 Y : \lambda Z : \lambda^2 Z^2)$ for all nonzero λ. The negative of $(X : Y : Z : Z^2)$ is $(X : -Y : Z : Z^2)$. The neutral element is represented as $(1 : 0 : 0)$. The group operation is directly related to that in affine Weierstrass form. Doubling is sped up considerably: it is computed as the composition of a 2-isogeny and its dual.

Fast doubling formulas for these curves, and mixed-addition formulas, were introduced by Doche, Icart, and Kohel in [**16**, Section 3.1]. We found some $\mathbf{S}-\mathbf{M}$ tradeoffs in these formulas and included the improved formulas in the EFD. We also developed general addition formulas and included those in the EFD. In the tables and graphs we denote this system by "2DIK."

Tripling-oriented Doche/Icart/Kohel curves. A point (x, y) on an elliptic curve $y^2 = x^3 + 3a(x + 1)^2$ is represented as $(X : Y : Z : Z^2)$ satisfying $Y^2 = X^3 + 3aZ^2(X + Z^2)^2$ and $(x, y) = (X/Z^2, Y/Z^3)$. Here $(X : Y : Z : Z^2) = (\lambda^2 X : \lambda^3 Y : \lambda Z : \lambda^2 Z^2)$ for all nonzero λ. The negative of $(X : Y : Z : Z^2)$ is $(X : -Y : Z : Z^2)$. The neutral element is represented as $(1 : 1 : 0 : 0)$. The group operation is directly related to that in affine Weierstrass form. Tripling is sped up considerably through the use of isogenies of degree 3.

Fast tripling formulas for these curves, and mixed-addition formulas, were introduced by Doche, Icart, and Kohel in [**16**, Section 3.2]. The doubling formulas in [**16**] are incorrect; corrected formulas appear in our paper [**2**, Section 2] with Birkner and Peters. We also found an $\mathbf{S}-\mathbf{M}$ tradeoff and derived formulas for general additions. In the tables and graphs we denote this system by "3DIK."

Montgomery coordinates. A point (x, y) on an elliptic curve $by^2 = x^3 + ax^2 + x$ is represented as $(X : Z)$ satisfying $x = X/Z$.

This representation is not compatible with addition and is not included in our tables. It loses information: observe that $(X : Z)$ also represents the point $(x, -y)$. However, this representation *does* allow scalar multiplication $P \mapsto nP$. Formulas introduced by Montgomery in [**28**, Section 10.3.1] use only $5\mathbf{M} + 4\mathbf{S}$ for each bit of n. In the graphs we denote this system by "Montgomery."

Jacobi intersections. A point (s, c, d) on an elliptic curve $s^2 + c^2 = 1$, $as^2 + d^2 = 1$ is represented as $(S : C : D : Z)$ satisfying $S^2 + C^2 = Z^2$, $aS^2 + D^2 = Z^2$ and $(s, c, d) = (S/Z, C/Z, D/Z)$. Here $(S : C : D : Z) = (\lambda S : \lambda C : \lambda D : \lambda Z)$ for all nonzero λ. The negative of $(S : C : D : Z)$ is $(-S : C : D : Z)$. The neutral element $(0, 1, 1)$ is represented as $(0 : 1 : 1 : 1)$.

The Jacobi intersection of $s^2 + c^2 = 1, as^2 + d^2 = 1$ is birationally equivalent to the Weierstrass-form elliptic curve $y^2 = x^3 + (2 - a)x^2 + (1 - a)x$. A typical point (s, c, d) on the Jacobi intersection corresponds to the point (x, y) on the Weierstrass curve defined by $x = (d - 1)(1 - a)/p$ and $y = s(1 - a)a/p$ where $p = ca - d + 1 - a$.

Chudnovsky and Chudnovsky in [**10**, formulas (4.9i) and (4.9ii)] presented fast doubling and addition formulas for Jacobi intersections. Liardet and Smart in [**26**] presented faster formulas. Slightly faster formulas, with an $\mathbf{S} - \mathbf{M}$ tradeoff, appear in the EFD. In the tables and graphs we denote this system by "JacIntersect."

Jacobi quartics. A point (x, y) on an elliptic curve $y^2 = x^4 + 2ax^2 + 1$ is represented as $(X : Y : Z)$ satisfying $Y^2 = X^4 + 2aX^2Z^2 + Z^4$ and $(x, y) = (X/Z, Y/Z^2)$. Here $(X : Y : Z) = (\lambda X : \lambda^2 Y : \lambda Z)$ for all nonzero λ. The negative of $(X : Y : Z)$ is $(-X : Y : Z)$. The neutral element $(0, 1)$ is represented as $(0 : 1 : 1)$.

The Jacobi-quartic elliptic curve $y^2 = x^4 + 2ax^2 + 1$ is birationally equivalent to the Weierstrass-form elliptic curve $2v^2 = u^3 - 2au^2 + (a^2 - 1)u$. A typical point (x, y) on the Jacobi quartic corresponds to the point (u, v) on the Weierstrass curve defined by $u = a + (y + 1)/x^2$ and $v = u/x$.

Billet and Joye in [**6**, Section 3], citing Whittaker and Watson [**34**], presented addition formulas and doubling formulas for Jacobi quartics. (Chudnovsky and Chudnovsky in [**10**, formulas (4.10i) et seq.] had stated formulas for computing on curves of this form, but they placed the neutral element at infinity and obtained slower formulas.) A flurry of speedups have been introduced this year by Duquesne in [**18**], by Hisil, Carter, and Dawson in [**22**], and by Feng and Wu in unpublished work.

Extended coordinates $(X : Y : Z : X^2 : 2XZ : Z^2 : X^2 + Z^2)$ save time. This was pointed out by Duquesne, modulo minor issues such as the 2 in $2XZ$. The speed records in the EFD for these coordinates are from Hisil, Carter, and Dawson for doubling and from Duquesne for addition, with an $\mathbf{S} - \mathbf{M}$ tradeoff from us. In the tables and graphs we denote this system by "ExtJQuartic."

For the original coordinates $(X : Y : Z)$, the speed records in the EFD are from Feng and Wu for doubling and from Billet and Joye for addition. In the tables and graphs we denote this system by "JQuartic."

Hessian curves. A point (x, y) on an elliptic curve $x^3 + y^3 + 1 = 3dxy$ is represented as $(X : Y : Z)$ satisfying $X^3 + Y^3 + Z^3 = 3dXYZ$ and $(x, y) = (X/Z, Y/Z)$. Here $(X : Y : Z) = (\lambda X : \lambda Y : \lambda Z)$ for all nonzero λ. The negative of $(X : Y : Z)$ is $(Y : X : Z)$. The neutral element is represented as $(1 : -1 : 0)$.

A Hessian-form elliptic curve $x^3 + y^3 + 1 = 3dxy$ is birationally equivalent to the Weierstrass-form elliptic curve $v^2 = u^3 - 27d(d^3 + 8)u + 54(d^6 - 20d^3 - 8)$. A typical point (x, y) on the Hessian curve corresponds to the point (u, v) on the Weierstrass curve defined by $u = p - 9d^2$ and $v = 3p(y - x)$ where $p = 12(d^3 - 1)/(d + x + y)$.

Chudnovsky and Chudnovsky in [**10**, formulas (4.20) et seq.] study the Hessian form for elliptic curves and give explicit formulas, which they credit to Cauchy and

Sylvester. Hisil, Carter, and Dawson in [**22**] give faster formulas for doublings. In the tables and graphs we denote this system by "Hessian."

Edwards curves. A point (x, y) on an elliptic curve $x^2 + y^2 = 1 + dx^2y^2$ is represented as $(X : Y : Z)$ satisfying $(X^2 + Y^2)Z^2 = Z^4 + dX^2Y^2$ and $(x, y) = (X/Z, Y/Z)$. Here $(X : Y : Z) = (\lambda X : \lambda Y : \lambda Z)$ for all nonzero λ. The negative of $(X : Y : Z)$ is $(-X : Y : Z)$. The neutral element $(0, 1)$ is represented as $(0 : 1 : 1)$.

The Edwards-form elliptic curve $x^2+y^2 = 1+dx^2y^2$ is birationally equivalent to the Montgomery-form elliptic curve $(1/e)v^2 = u^3 + (4/e-2)u^2 + u$ where $e = 1-d$. A point (x, y) on the Edwards curve, with nonzero x, corresponds to the point (u, v) on the Montgomery curve defined by $u = (1 + y)/(1 - y)$ and $v = 2u/x$.

This normal form for elliptic curves was introduced by Edwards in [**19**], generalizing from one curve studied by Euler and Gauss. The Edwards addition law in affine coordinates is given by

$$(x_1, y_1) + (x_2, y_2) = \left(\frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2}, \frac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2}\right).$$

Our paper [**4**] studied the projective version and introduced fast explicit formulas for all the group operations used in this paper. In the tables and graphs we denote this system by "Edwards."

Inverted Edwards coordinates. A point (x, y) on an elliptic curve $x^2+y^2 = 1 + dx^2y^2$ is represented as $(X : Y : Z)$ satisfying $(X^2 + Y^2)Z^2 = X^2Y^2 + dZ^4$ and $(x, y) = (Z/X, Z/Y)$. Here $(X : Y : Z) = (\lambda X : \lambda Y : \lambda Z)$ for all nonzero λ. This representation does not cover the points $(0, \pm 1)$ and $(\pm 1, 0)$. The negative of $(X : Y : Z)$ is $(-X : Y : Z)$. The neutral element $(0, 1)$ is represented by $(1, 0, 0)$.

We introduced inverted Edwards coordinates in [**5**] as a different coordinate system for Edwards curves. In the tables and graphs we denote this system by "InvEdwards."

3. Fast scalar multiplication

This section defines, for each positive integer n and each $m \in \{3, 5, 7, 9, \ldots\}$, a particular "addition-subtraction chain" $C_m(n)$. This chain can be viewed as a function that computes nP given a point P. The extra parameter m is the "maximum precomputed multiple" in $C_m(n)$; if m is chosen sensibly then $C_m(n)$ has very few additions, subtractions, and doublings.

For each $m \in \{3, 5, \ldots, 39\}$ and each $\ell \in \{160, 256, 512\}$, our software averages the cost of this chain $C_m(n)$ for 100000 uniform random ℓ-bit integers n, and then identifies the choice of m that minimizes this average. The optimum m depends not only on ℓ but also on the definition of "cost." In particular, Table 4.1, Table 4.2, and Table 4.3 show that the optimum m depends on the choice of inversion strategy discussed in Section 4 and on the elliptic-curve coordinate system discussed in Section 2.

This chain $C_m(n)$ is a state-of-the-art combination of

- the "window" idea introduced by Brauer in [**8**],
- the "sliding window" idea introduced by Thurber in [**33**],
- the obvious "signed window" idea for groups where subtraction is as easy as addition,
- the "fractional window" idea introduced by Möller in [**27**, Section 5], and

- a few minor tweaks.

We do not claim any novelty for the ideas here. Our goal in this section is to state for the record—and for lack of a suitable reference—what we did in our experiments.

We are not aware of any noticeably better chains using additions, subtractions, and doublings. For some coordinate systems there are better chains using additions, subtractions, doublings, and triplings; these "double-base" chains with precomputations were introduced by Doche and Imbert in [**17**], building on an idea by Dimitrov, Imbert, and Mishra in [**14**], and were further improved in our recent paper [**2**] with Birkner and Peters. The optimizations discussed in Section 4 of this paper can be applied to the best chains identified in [**2**].

Definition of $C_m(n)$: the typical cases. If n is even, and not covered by the base cases discussed below, then $C_m(n)$ is the chain

$$C_m(n/2); n = 2(n/2).$$

In other words, we compute $(n/2)P$ recursively and then double $(n/2)P$ to obtain nP.

If n is odd, and not covered by the base cases discussed below, then $C_m(n)$ is the chain

$$C_m(n-r); n = (n-r) + (r).$$

Here $r \in \{-m, \ldots, -3, -1, 1, 3, \ldots, m-2, m\}$ is chosen to maximize the maximal power of 2 dividing $n - r$. (The base cases guarantee that $C_m(n-r)$ contains $1, 3, \ldots, m-2, m$. See below.) In other words, we compute $(n-r)P$ recursively and then add rP to it to obtain nP. If r is negative then we actually subtract $(-r)P$ from $(n-r)P$; from now on we ignore the distinction between subtractions and additions.

Definition of $C_m(n)$: the base cases. If $n \in \{1, 2, 3, 5, \ldots, m-2, m\}$ then $C_m(n)$ is the chain

$$1; 2 = 2(1); 3 = (2) + (1); 5 = (3) + (2); \ldots; m = (m-2) + (2).$$

These $(m+1)/2$ additions are called **precomputations**; every $C_m(n)$ starts with the same precomputations.

If $n = m + 2$ then $C_m(n)$ is the chain

$$C_m(m); n = (m) + (2).$$

If $m + 4 \le n \le 3m - 2$ and $n \bmod 6 = 1$ then $C_m(n)$ is the chain

$$C_m(m); \frac{2n+4}{3} = 2\left(\frac{n+2}{3}\right); n = \left(\frac{2n+4}{3}\right) + \left(\frac{n-4}{3}\right).$$

If $m + 4 \le n \le 3m$ and $n \bmod 6 = 3$ then $C_m(n)$ is the chain

$$C_m(m); \frac{2n}{3} = 2\left(\frac{n}{3}\right); n = \left(\frac{2n}{3}\right) + \left(\frac{n}{3}\right).$$

This case could be computed more efficiently with dedicated triplings.

If $m + 4 \le n \le 3m - 4$ and $n \bmod 6 = 5$ then $C_m(n)$ is the chain

$$C_m(m); \frac{2n-4}{3} = 2\left(\frac{n-2}{3}\right); n = \left(\frac{2n-4}{3}\right) + \left(\frac{n+4}{3}\right).$$

Finally, if $4 \le n \le 2m-2$ and $n \bmod 4 = 0$ then $C_m(n)$ is the chain

$$C_m(m); n = \left(\frac{n-2}{2}\right) + \left(\frac{n+2}{2}\right).$$

It might seem easier to obtain n as $2(n/2)$, but both $n/2-1$ and $n/2+1$ are in $C_m(m)$ while $n/2$ is not; the cost of obtaining $n/2$ generally outweighs the difference in costs between an addition and a doubling.

A numerical example. The chain $C_5(314159)$ is

$$\begin{aligned}
&1;\quad 2=2(1);\quad 3=(2)+(1);\quad 5=(3)+(2);\quad 10=2(5);\\
&20=2(10);\quad 40=2(20);\quad 80=2(40);\quad 77=(80)-(3);\\
&154=2(77);\quad 308=2(154);\quad 616=2(308);\quad 1232=2(616);\\
&1227=(1232)-(5);\quad 2454=2(1227);\quad 4908=2(2454);\\
&9816=2(4908);\quad 19632=2(9816);\quad 19635=(19632)+(3);\\
&39270=2(19635);\quad 78540=2(39270);\quad 157080=2(78540);\\
&314160=2(157080);\quad 314159=(314160)-(1).
\end{aligned}$$

This chain specifies a computation of $314159P$ from P with 6 additions and 17 doublings. The precomputation doubles P to obtain $2P$, then adds P to $2P$ to obtain $3P$, then adds $2P$ to $3P$ to obtain $5P$. The main computation doubles $5P$ to obtain $10P$, doubles $10P$ to obtain $20P$, etc.

4. Using inversions

The chain $C_m(n)$ defined in Section 3 specifies a computation of nP that begins by precomputing $2P, 3P, 5P, \ldots, mP$. This section explains several different strategies for precomputing $2P, 3P, 5P, \ldots, mP$.

We always assume that P is given in affine form (Z-coordinate 1) in the coordinate system under consideration. Additions involving P are then mixed additions. Additions involving the other precomputed points are not mixed additions — unless we use inversions to convert those points to affine form.

No inversions. The simplest strategy is to compute $2P$ by doubling P, then $3P$ by adding $2P$ to P, then $5P$ by readding $2P$ to $3P$, then $7P$ by readding $2P$ to $5P$, then $9P$ by readding $2P$ to $7P$, etc., without any inversions. This strategy involves 1 mDBL, 1 mADD, and $(m-3)/2$ reADD.

The rest of the scalar multiplication frequently adds these precomputed points $P, 3P, 5P, \ldots, mP$ to other points. Each addition involving P is 1 mADD; the other additions are reADD, except that the first addition involving mP is usually an ADD.

For example, for $\ell = 512$ and $m = 29$, we tried 100000 uniform random integers $n \in \{0, 1, \ldots, 2^{512}-1\}$, and found that $C_m(n)$ used on average 1 mDBL, 505.50 DBL, 10.22 mADD, 0.90 ADD, and 77.41 reADD.

One inversion. The "invert $\{m\}$" strategy is to compute $2P, 3P, 5P, \ldots, mP$ exactly as above, and then convert $3P, 5P, \ldots, mP$ to affine coordinates.

The cost of this conversion is 1 SCALE and $(m-3)/2$ xSCALE. The benefit of this conversion is that the subsequent additions involving $3P, 5P, \ldots, mP$ are mADD instead of reADD.

For example, for $\ell = 512$ and $m = 29$, we found that $C_m(n)$ used on average 1 SCALE, 13 xSCALE, 1.36 mDBL, 505.14 DBL, 0.62 mmADD, 74.92 mADD, and 13 reADD. The benefit in this case is, approximately, that 64 reADD were replaced by 64 mADD; this benefit outweighs the cost of 1 SCALE and 13 xSCALE if $\mathbf{I}/\mathbf{M}$ is small.

The exact $\mathbf{I}/\mathbf{M}$ break-even point, the point below which invert-$\{m\}$ speeds up the scalar multiplication, depends on the coordinate system and on ℓ. Figure 1.3 shows the number of multiplications per bit for $\ell = 512$ in all systems considered in this paper. The right-most bend of each graph happens when $\mathbf{I}/\mathbf{M}$ is at the break-even point. The exact data can be found in Table 4.3 which also states the optimal value of m. Examples: For 2DIK the benefits of mADD over reADD are so large that for $1\mathbf{I} \leq 263.2\mathbf{M}$ one inversion should be used to scale the precomputed points. For Hessian the break-even point is at $1\mathbf{I} = 72\mathbf{M}$. For Edwards, for all realistic $\mathbf{I}/\mathbf{M}$ ratios, even one inversion is not worthwhile.

Dahmen, Okeya, and Schepers in [**13**] proposed a different strategy to compute $3P, 5P, \ldots, mP$ in affine coordinates using a total of $1\mathbf{I}+(5m-6)\mathbf{M}+(2m+2)\mathbf{S}$. We use this "DOS" strategy for Jacobian and Jacobian-3, as in [**13**], and for Projective and Projective-3. We have not yet explored the extent to which similar strategies can be used for other coordinate systems.

DOS reduces the cost of obtaining the precomputed points in affine coordinates. This is reflected in our results. Consider, for example, the graphs in Figure 1.3 for $\ell = 512$. The graph for Jacobian (which uses DOS) has its right-most bend at a much higher $\mathbf{I}/\mathbf{M}$ ratio than that of Jacobian-O (which uses invert-$\{m\}$). This implies that the number of multiplications per bit is decreasing with decreasing $\mathbf{I}$ for $1\mathbf{I} \leq 226.8\mathbf{M}$ while this effect happens for Jacobian-O only for $1\mathbf{I} \leq 131.1\mathbf{M}$. The same observation holds for Jacobian-3, Jacobian-3-O, Projective, and Projective-3.

Two inversions. The "invert $\{2, m\}$" strategy computes $2P$; converts $2P$ to affine coordinates; computes $3P, 5P, \ldots, mP$; and converts $3P, 5P, \ldots, mP$ to affine coordinates.

The cost of the conversion of $2P$ to affine coordinates is 1 SCALE. The benefit of this conversion is that the 1 mADD and $(m-3)/2$ reADD to compute $3P, 5P, \ldots, mP$ are replaced by 1 mmADD and $(m-3)/2$ mADD.

For example, for $\ell = 512$ and $m = 29$, we found that $C_m(n)$ used on average 2 SCALE, 13 xSCALE, 1.36 mDBL, 505.14 DBL, 1.64 mmADD, and 86.92 mADD. The benefit of invert-$\{2, m\}$ over invert-$\{m\}$ is, approximately, that 13 reADD were replaced by 13 mADD; this benefit outweighs the cost of 1 SCALE if $\mathbf{I}/\mathbf{M}$ is very small.

The break-even point where invert-$\{2, m\}$ speeds up the scalar multiplication again depends on the coordinate system and ℓ. Consider, for example, Figure 1.3 with $\ell = 512$. The 2DIK graph is horizontal at the right side of the figure, bends downwards when $\mathbf{I}/\mathbf{M}$ drops to 263.2, and bends downwards again when $\mathbf{I}/\mathbf{M}$ drops to 40.1; this second bend happens when $\mathbf{I}/\mathbf{M}$ is small enough to reach the break-even point between invert-$\{m\}$ and invert-$\{2, m\}$. For Hessian the break-even point is at $1\mathbf{I} = 20.4\mathbf{M}$. For all realistic $\mathbf{I}/\mathbf{M}$ ratios invert-$\{2, m\}$ is never better than DOS.

Three inversions. The "invert $\{2, 2\lfloor(m+1)/4\rfloor, m\}$" strategy first computes $2P$; converts $2P$ to affine coordinates; computes $3P, 5P, \ldots, (2c-1)P$ where $c =$

$\lfloor (m+1)/4 \rfloor$; uses an extra addition to compute $2cP$; converts $3P, 5P, \ldots, (2c-1)P$, $2cP$ to affine coordinates; computes $(2c+1)P, (2c+3)P, \ldots, (4c-1)P$ by adding $P, 3P, \ldots, (2c-1)P$ to $2cP$; computes mP if $m = 4c+1$; and converts $(2c+1)P$, $(2c+3)P, \ldots, mP$ to affine coordinates.

The cost of this strategy, compared to the previous strategy, is 1 mADD to compute $2cP$, plus 1 SCALE for the intermediate conversion to affine coordinates. (Sometimes the 1 mADD can be replaced by 1 DBL; we have not included this tweak yet.) The benefit of this conversion is that c mADD for the computation of $(2c+1)P, (2c+3)P, \ldots, (4c-1)P$ are replaced by c mmADD.

For example, for $\ell = 512$ and $m = 29$, we found that $C_m(n)$ used on average 3 SCALE, 13 xSCALE, 1.36 mDBL, 505.14 DBL, 8.64 mmADD, and 80.90 mADD. The benefit in this case is, approximately, that 7 mADD were replaced by 7 mmADD; this benefit can outweigh the cost of 1 mADD and 1 SCALE if $\mathbf{I}/\mathbf{M}$ is extremely small.

Only 2DIK and 3DIK profit from invert-$\{2, 2\lfloor (m+1)/4 \rfloor, m\}$ for $\ell = 512$ and only for very small $\mathbf{I}/\mathbf{M}$-ratios. For $\ell = 256$ or smaller no system benefits from invert-$\{2, 2\lfloor (m+1)/4 \rfloor, m\}$. For larger values of ℓ this strategy becomes more interesting.

More inversions. One could articulate a further strategy using 4 inversions, e.g., "invert-$\{2, 2\lfloor (m+1)/8 \rfloor, 2\lfloor (m+1)/4 \rfloor, m\}$" that computes $2P$; converts $2P$ to affine coordinates; computes $3P, 5P, \ldots, (2b-1)P$ where $b = \lfloor (m+1)/8 \rfloor$; uses an extra addition to compute $2bP$; etc. The extra benefit here is half as large as the benefit of invert-$\{2, 2\lfloor (m+1)/4 \rfloor, m\}$ over invert-$\{2, m\}$, while the extra cost is just as large, but if ℓ is gigantic then the benefit outweighs the cost.

When $m+1$ is a power of 2, adding more and more inversions in the same way eventually produces the "invert-$\{2, 4, 8, 16, \ldots, m\}$" strategy introduced by Cohen, Miyaji, and Ono in [**12**].

History. As far as we know, the first study of scalar multiplications including the costs of the precomputations was done by Cohen, Miyaji, and Ono in [**12**]. They consider only two options for the precomputations: the no-inversion strategy and their new strategy using $\lg(m+1)$ inversions. The invert-$\{m\}$ strategy is considered by Doche and Lange in [**15**]. The first mention of invert-$\{2, m\}$ that we could find was by Elmegaard-Fessel in [**20**]. The DOS strategy was developed by Dahmen, Okeya, and Schepers in [**13**]; it is better than invert-$\{m\}$ whenever it is applicable.

The invert-$\{2, 2\lfloor (m+1)/4 \rfloor, m\}$ strategy that we presented for 3 inversions is based on extensive computer experiments: we allowed the computer to systematically explore a much wider space of strategies and to identify the best strategies. We have not located this strategy, or any other strategies between 3 and $\lg(m+1) - 1$ inversions, in the literature.

References

[1] Rana Barua, Tanja Lange (editors), *Progress in Cryptology — INDOCRYPT 2006, 7th International Conference on Cryptology in India, Kolkata, India, December 11–13, 2006, Proceedings*, Lecture Notes in Computer Science, 4329, Springer, 2006. ISBN 3–540–49767–6. See [17].

[2] Daniel J. Bernstein, Peter Birkner, Tanja Lange, Christiane Peters, *Optimizing double-base elliptic-curve single-scalar multiplication*, in [**32**] (2007), 167-182. Citations in this document: §2, §2, §3, §3.

[3] Daniel J. Bernstein, Tanja Lange, *Explicit-formulas database* (2007). URL: `http://hyperelliptic.org/EFD`. Citations in this document: §2, §2.

[4] Daniel J. Bernstein, Tanja Lange, *Faster addition and doubling on elliptic curves*, in [**24**] (2007), 29–50. Citations in this document: §2.

[5] Daniel J. Bernstein, Tanja Lange, *Inverted Edwards coordinates*, in [**7**] (2007), 20–27. Citations in this document: §2.

[6] Olivier Billet, Marc Joye, *The Jacobi model of an elliptic curve and side-channel analysis*, in [**21**] (2003), 34–42. MR 2005c:94045. URL: `http://eprint.iacr.org/2002/125`. Citations in this document: §2.

[7] Serdar Boztas, Hsiao-Feng Lu (editors), *Applied algebra, algebraic algorithms and error-correcting codes*, Lecture Notes in Computer Science, 4851, Springer, 2007. See [5].

[8] Alfred Brauer, *On addition chains*, Bulletin of the American Mathematical Society **45** (1939), 736–739. ISSN 0273–0979. MR 1,40a. URL: `http://cr.yp.to/bib/entries.html#1939/brauer`. Citations in this document: §3.

[9] Éric Brier, Marc Joye, *Fast point multiplication on elliptic curves through isogenies*, in [**21**] (2003), 43–50. Citations in this document: §2.

[10] David V. Chudnovsky, Gregory V. Chudnovsky, *Sequences of numbers generated by addition in formal groups and new primality and factorization tests*, Advances in Applied Mathematics **7** (1986), 385–434. MR 88h:11094. Citations in this document: §2, §2, §2, §2, §2.

[11] Henri Cohen, Gerhard Frey (editors), *Handbook of elliptic and hyperelliptic curve cryptography*, CRC Press, 2005. ISBN 1–58488–518–1. MR 2007f:14020. See [15].

[12] Henri Cohen, Atsuko Miyaji, Takatoshi Ono, *Efficient elliptic curve exponentiation using mixed coordinates*, in [**29**] (1998), 51–65. Citations in this document: §1, §1, §1, §2, §4, §4.

[13] Erik Dahmen, Katsuyuki Okeya, Daniel Schepers, *Affine precomputation with sole inversion in elliptic curve cryptography*, in [**30**] (2007), 245–258. Citations in this document: §1, §4, §4, §4.

[14] Vassil Dimitrov, Laurent Imbert, Pradeep K. Mishra., *Efficient and secure elliptic curve point multiplication using double-base chains*, in [**31**] (2005), 59–78. Citations in this document: §3.

[15] Christophe Doche, *Exponentiation*, in [**11**] (2005), 145–168. MR 2162725. Citations in this document: §4.

[16] Christophe Doche, Thomas Icart, David R. Kohel, *Efficient scalar multiplication by isogeny decompositions*, in [**35**] (2006), 191–206. Citations in this document: §2, §2, §2.

[17] Christophe Doche, Laurent Imbert, *Extended double-base number system with applications to elliptic curve cryptography*, in [**1**] (2006), 335–348. Citations in this document: §3.

[18] Sylvain Duquesne, *Improving the arithmetic of elliptic curves in the Jacobi model*, Information Processing Letters **104** (2007), 101–105. Citations in this document: §2.

[19] Harold M. Edwards, *A normal form for elliptic curves*, Bulletin of the American Mathematical Society **44** (2007), 393–422. URL: `http://www.ams.org/bull/2007-44-03/S0273-0979-07-01153-6/home.html`. Citations in this document: §2.

[20] Lars Elmegaard-Fessel, *Efficient scalar multiplication and security against power analysis in cryptosystems based on the NIST elliptic curves over prime fields* (2006). URL: `http://eprint.iacr.org/2006/313`. Citations in this document: §4.

[21] Marc Fossorier, Tom Hoeholdt, Alain Poli (editors), *Applied algebra, algebraic algorithms and error-correcting codes*, Lecture Notes in Computer Science, 2643, Springer, 2003. ISBN 3–540–40111–3. MR 2004j:94001. See [6], [9].

[22] Huseyin Hisil, Gary Carter, Ed Dawson, *New formulae for efficient elliptic curve arithmetic*, in [**32**] (2007). Citations in this document: §2, §2.

[23] Çetin Kaya Koç, David Naccache, Christof Paar (editors), *Cryptographic hardware and embedded systems — CHES 2001, third international workshop, Paris, France, May 14–16, 2001, proceedings*, Lecture Notes in Computer Science, 2162, Springer, 2001. ISBN 3–540–42521–7. MR 2003g:94002. See [26].

[24] Kaoru Kurosawa (editor), *Advances in cryptology — ASIACRYPT 2007*, Lecture Notes in Computer Science, 4833, Springer, 2007. See [4].

[25] Pil Joong Lee, Chae Hoon Lim (editors), *Information security and cryptology — ICISC 2002*, Lecture Notes in Computer Science, 2587, Springer, 2003. ISBN 3–540–00716–4. See [27].

[26] Pierre-Yvan Liardet, Nigel P. Smart, *Preventing SPA/DPA in ECC systems using the Jacobi form*, in [**23**] (2001), 391–401. MR 2003k:94033. Citations in this document: §2.

[27] Bodo Möller, *Improved techniques for fast exponentiation*, in [**25**] (2003), 298–312. Citations in this document: §3.

[28] Peter L. Montgomery, *Speeding the Pollard and elliptic curve methods of factorization*, Mathematics of Computation **48** (1987), 243–264. ISSN 0025–5718. MR 88e:11130. URL: http://links.jstor.org/sici?sici=0025-5718(198701)48:177<243:STPAEC>2.0.CO;2-3. Citations in this document: §2.

[29] Kazuo Ohta, Dingyi Pei (editors), *Advances in cryptology — ASIACRYPT'98: international conference on the theory and application of cryptology and information security, Beijing, China, October 18–22, 1998, proceedings*, Lecture Notes in Computer Science, 1514, Springer-Verlag, Berlin, 1998. ISBN 3–540–65109–8. MR 2000h:94002. See [12].

[30] Josef Pieprzyk, Hossein Ghodosi, Ed Dawson (editors), *Information security and privacy, 12th Australasian conference, ACISP 2007, Townsville, Australia, July 2-4, 2007, proceedings*, Lecture Notes in Computer Science, 4586, Springer, 2007. ISBN 978–3–540–73457–4. See [13].

[31] Bimal Roy (editor), *Advances in Cryptology — ASIACRYPT 2005, 11th International Conference on the Theory and Application of Cryptology and Information Security, Chennai, India, December 4–8, 2005, Proceedings*, Lecture Notes in Computer Science, 3788, Springer, Berlin, 2005. See [14].

[32] Kannan Srinathan, C. Pandu Rangan, Moti Yung (editors), *Indocrypt 2007*, Lecture Notes in Computer Science, 4859, Springer, 2007. See [2], [22].

[33] Edward G. Thurber, *On addition chains* $l(mn) \le l(n) - b$ *and lower bounds for* $c(r)$, Duke Mathematical Journal **40** (1973), 907–913. ISSN 0012–7094. MR 48:8429. URL: http://cr.yp.to/bib/entries.html#1973/thurber. Citations in this document: §3.

[34] E. T. Whittaker, G. N. Watson (editors), *A course of modern analysis*, Cambridge University Press, Cambridge, 1927. ISBN 0–521–58807–3. MR 97k:01072. Citations in this document: §2.

[35] Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, Tal Malkin (editors), *9th international conference on theory and practice in public-key cryptography, New York, NY, USA, April 24–26, 2006, proceedings*, Lecture Notes in Computer Science, 3958, Springer, Berlin, 2006. ISBN 978–3–540–33851–2. See [16].

Department of Mathematics, Statistics, and Computer Science (M/C 249), The University of Illinois at Chicago, Chicago, IL 60607–7045

E-mail address: djb@cr.yp.to

Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, P.O. Box 513, 5600 MB Eindhoven, Netherlands

E-mail address: tanja@hyperelliptic.org

TABLE 4.1. Optimal parameters for 160-bit scalars for each curve shape and each inversion strategy, assuming $\mathbf{S}/\mathbf{M} = 0.8$.

Bits	Curve shape	Inversions	m	Multiplications and squarings	$\mathbf{I}/\mathbf{M}$
160	2DIK	0**I**	13	$684.8\mathbf{M} + 939.6\mathbf{S} \approx 1436.5\mathbf{M}$	large
160	2DIK	1**I**	7	$609.4\mathbf{M} + 927.7\mathbf{S} \approx 1351.6\mathbf{M}$	≤ 84.9
160	2DIK	2**I**	9	$598.9\mathbf{M} + 923.4\mathbf{S} \approx 1337.6\mathbf{M}$	≤ 14.0
160	2DIK	3**I**	9	$593.9\mathbf{M} + 929.4\mathbf{S} \approx 1337.5\mathbf{M}$	
160	3DIK	0**I**	13	$625.8\mathbf{M} + 1275.0\mathbf{S} \approx 1645.8\mathbf{M}$	large
160	3DIK	1**I**	7	$576.0\mathbf{M} + 1238.9\mathbf{S} \approx 1567.1\mathbf{M}$	≤ 78.7
160	3DIK	2**I**	9	$571.0\mathbf{M} + 1229.7\mathbf{S} \approx 1554.8\mathbf{M}$	≤ 12.3
160	3DIK	3**I**	9	$569.1\mathbf{M} + 1231.7\mathbf{S} \approx 1554.5\mathbf{M}$	
160	Edwards	0**I**	13	$797.2\mathbf{M} + 655.5\mathbf{S} \approx 1321.6\mathbf{M}$	large
160	Edwards	1**I**	9	$796.5\mathbf{M} + 657.6\mathbf{S} \approx 1322.6\mathbf{M}$	
160	Edwards	2**I**	9	$792.4\mathbf{M} + 657.6\mathbf{S} \approx 1318.5\mathbf{M}$	
160	Edwards	3**I**	9	$791.5\mathbf{M} + 660.6\mathbf{S} \approx 1320.0\mathbf{M}$	
160	ExtJQuartic	0**I**	13	$727.6\mathbf{M} + 726.1\mathbf{S} \approx 1308.5\mathbf{M}$	large
160	ExtJQuartic	1**I**	9	$725.3\mathbf{M} + 735.3\mathbf{S} \approx 1313.5\mathbf{M}$	
160	ExtJQuartic	2**I**	9	$722.2\mathbf{M} + 737.3\mathbf{S} \approx 1312.1\mathbf{M}$	
160	ExtJQuartic	3**I**	9	$721.3\mathbf{M} + 746.3\mathbf{S} \approx 1318.3\mathbf{M}$	
160	Hessian	0**I**	13	$1475.7\mathbf{M} + 157.7\mathbf{S} \approx 1601.9\mathbf{M}$	large
160	Hessian	1**I**	9	$1453.2\mathbf{M} + 159.1\mathbf{S} \approx 1580.5\mathbf{M}$	≤ 21.4
160	Hessian	2**I**	11	$1447.4\mathbf{M} + 158.8\mathbf{S} \approx 1574.4\mathbf{M}$	
160	Hessian	3**I**	9	$1448.2\mathbf{M} + 162.1\mathbf{S} \approx 1577.9\mathbf{M}$	
160	InvEdwards	0**I**	13	$763.4\mathbf{M} + 655.5\mathbf{S} \approx 1287.8\mathbf{M}$	large
160	InvEdwards	1**I**	9	$763.1\mathbf{M} + 657.1\mathbf{S} \approx 1288.8\mathbf{M}$	
160	InvEdwards	2**I**	9	$761.1\mathbf{M} + 656.0\mathbf{S} \approx 1285.9\mathbf{M}$	
160	InvEdwards	3**I**	9	$764.2\mathbf{M} + 657.1\mathbf{S} \approx 1289.8\mathbf{M}$	
160	JacIntersect	0**I**	15	$837.9\mathbf{M} + 689.3\mathbf{S} \approx 1389.4\mathbf{M}$	large
160	JacIntersect	1**I**	9	$864.8\mathbf{M} + 693.5\mathbf{S} \approx 1419.6\mathbf{M}$	
160	JacIntersect	2**I**	9	$864.7\mathbf{M} + 693.4\mathbf{S} \approx 1419.5\mathbf{M}$	
160	JacIntersect	3**I**	9	$863.8\mathbf{M} + 697.5\mathbf{S} \approx 1421.8\mathbf{M}$	
160	Jacobian	0**I**	13	$471.1\mathbf{M} + 1378.4\mathbf{S} \approx 1573.8\mathbf{M}$	large
160	Jacobian	1**I**	15	$402.7\mathbf{M} + 1366.4\mathbf{S} \approx 1495.8\mathbf{M}$	≤ 78.0
160	Jacobian	2**I**	9	$416.3\mathbf{M} + 1384.4\mathbf{S} \approx 1523.8\mathbf{M}$	
160	Jacobian	3**I**	9	$414.4\mathbf{M} + 1386.4\mathbf{S} \approx 1523.5\mathbf{M}$	
160	Jacobian-3	0**I**	13	$780.4\mathbf{M} + 914.3\mathbf{S} \approx 1511.9\mathbf{M}$	large
160	Jacobian-3	1**I**	15	$710.9\mathbf{M} + 904.1\mathbf{S} \approx 1434.1\mathbf{M}$	≤ 77.8
160	Jacobian-3	2**I**	9	$725.7\mathbf{M} + 920.3\mathbf{S} \approx 1462.0\mathbf{M}$	
160	Jacobian-3	3**I**	9	$723.8\mathbf{M} + 922.4\mathbf{S} \approx 1461.6\mathbf{M}$	
160	JQuartic	0**I**	13	$607.5\mathbf{M} + 1033.5\mathbf{S} \approx 1434.2\mathbf{M}$	large
160	JQuartic	1**I**	9	$604.5\mathbf{M} + 1040.8\mathbf{S} \approx 1437.1\mathbf{M}$	
160	JQuartic	2**I**	9	$600.4\mathbf{M} + 1040.7\mathbf{S} \approx 1432.9\mathbf{M}$	
160	JQuartic	3**I**	9	$597.5\mathbf{M} + 1043.7\mathbf{S} \approx 1432.4\mathbf{M}$	
160	Projective	0**I**	13	$1158.6\mathbf{M} + 1000.7\mathbf{S} \approx 1959.2\mathbf{M}$	large
160	Projective	1**I**	15	$1078.7\mathbf{M} + 1012.0\mathbf{S} \approx 1888.2\mathbf{M}$	≤ 71.0
160	Projective	2**I**	13	$1102.3\mathbf{M} + 1000.2\mathbf{S} \approx 1902.5\mathbf{M}$	
160	Projective	3**I**	15	$1100.5\mathbf{M} + 1000.9\mathbf{S} \approx 1901.2\mathbf{M}$	

TABLE 4.2. Optimal parameters for 256-bit scalars for each curve shape and each inversion strategy, assuming $\mathbf{S}/\mathbf{M} = 0.8$.

Bits	Curve shape	Inversions	m	Multiplications and squarings	$\mathbf{I}/\mathbf{M}$
256	2DIK	0**I**	13	$1060.5\mathbf{M} + 1498.9\mathbf{S} \approx 2259.7\mathbf{M}$	large
256	2DIK	1**I**	13	$950.7\mathbf{M} + 1471.3\mathbf{S} \approx 2127.8\mathbf{M}$	≤ 131.9
256	2DIK	2**I**	15	$926.8\mathbf{M} + 1464.2\mathbf{S} \approx 2098.2\mathbf{M}$	≤ 29.6
256	2DIK	3**I**	15	$920.7\mathbf{M} + 1469.2\mathbf{S} \approx 2096.1\mathbf{M}$	
256	3DIK	0**I**	13	$972.3\mathbf{M} + 2038.8\mathbf{S} \approx 2603.3\mathbf{M}$	large
256	3DIK	1**I**	13	$901.7\mathbf{M} + 1975.8\mathbf{S} \approx 2482.4\mathbf{M}$	≤ 120.9
256	3DIK	2**I**	15	$886.7\mathbf{M} + 1961.2\mathbf{S} \approx 2455.6\mathbf{M}$	≤ 26.8
256	3DIK	3**I**	15	$884.6\mathbf{M} + 1958.2\mathbf{S} \approx 2451.2\mathbf{M}$	
256	Edwards	0**I**	17	$1245.6\mathbf{M} + 1053.6\mathbf{S} \approx 2088.5\mathbf{M}$	large
256	Edwards	1**I**	15	$1237.9\mathbf{M} + 1054.2\mathbf{S} \approx 2081.3\mathbf{M}$	
256	Edwards	2**I**	15	$1231.0\mathbf{M} + 1054.3\mathbf{S} \approx 2074.4\mathbf{M}$	
256	Edwards	3**I**	15	$1229.9\mathbf{M} + 1055.2\mathbf{S} \approx 2074.1\mathbf{M}$	
256	ExtJQuartic	0**I**	17	$1144.0\mathbf{M} + 1156.2\mathbf{S} \approx 2068.9\mathbf{M}$	large
256	ExtJQuartic	1**I**	15	$1136.4\mathbf{M} + 1165.4\mathbf{S} \approx 2068.7\mathbf{M}$	
256	ExtJQuartic	2**I**	15	$1130.4\mathbf{M} + 1167.4\mathbf{S} \approx 2064.3\mathbf{M}$	
256	ExtJQuartic	3**I**	15	$1131.4\mathbf{M} + 1175.4\mathbf{S} \approx 2071.7\mathbf{M}$	
256	Hessian	0**I**	17	$2339.9\mathbf{M} + 253.2\mathbf{S} \approx 2542.4\mathbf{M}$	large
256	Hessian	1**I**	15	$2294.6\mathbf{M} + 254.2\mathbf{S} \approx 2498.0\mathbf{M}$	≤ 44.4
256	Hessian	2**I**	15	$2282.7\mathbf{M} + 254.3\mathbf{S} \approx 2486.1\mathbf{M}$	≤ 11.9
256	Hessian	3**I**	15	$2286.6\mathbf{M} + 254.3\mathbf{S} \approx 2490.0\mathbf{M}$	
256	InvEdwards	0**I**	17	$1195.8\mathbf{M} + 1053.6\mathbf{S} \approx 2038.7\mathbf{M}$	large
256	InvEdwards	1**I**	15	$1189.4\mathbf{M} + 1053.6\mathbf{S} \approx 2032.3\mathbf{M}$	
256	InvEdwards	2**I**	15	$1184.5\mathbf{M} + 1052.6\mathbf{S} \approx 2026.6\mathbf{M}$	
256	InvEdwards	3**I**	15	$1190.4\mathbf{M} + 1049.6\mathbf{S} \approx 2030.1\mathbf{M}$	
256	JacIntersect	0**I**	17	$1301.9\mathbf{M} + 1104.4\mathbf{S} \approx 2185.4\mathbf{M}$	large
256	JacIntersect	1**I**	15	$1337.0\mathbf{M} + 1105.3\mathbf{S} \approx 2221.2\mathbf{M}$	
256	JacIntersect	2**I**	15	$1337.0\mathbf{M} + 1105.3\mathbf{S} \approx 2221.3\mathbf{M}$	
256	JacIntersect	3**I**	15	$1338.9\mathbf{M} + 1107.3\mathbf{S} \approx 2224.8\mathbf{M}$	
256	Jacobian	0**I**	13	$721.6\mathbf{M} + 2213.2\mathbf{S} \approx 2492.1\mathbf{M}$	large
256	Jacobian	1**I**	15	$610.6\mathbf{M} + 2198.3\mathbf{S} \approx 2369.2\mathbf{M}$	≤ 122.9
256	Jacobian	2**I**	15	$636.6\mathbf{M} + 2211.3\mathbf{S} \approx 2405.6\mathbf{M}$	
256	Jacobian	3**I**	15	$634.5\mathbf{M} + 2208.3\mathbf{S} \approx 2401.2\mathbf{M}$	
256	Jacobian-3	0**I**	13	$1222.9\mathbf{M} + 1461.1\mathbf{S} \approx 2391.8\mathbf{M}$	large
256	Jacobian-3	1**I**	15	$1110.8\mathbf{M} + 1448.0\mathbf{S} \approx 2269.2\mathbf{M}$	≤ 122.6
256	Jacobian-3	2**I**	15	$1136.8\mathbf{M} + 1461.0\mathbf{S} \approx 2305.6\mathbf{M}$	
256	Jacobian-3	3**I**	15	$1134.7\mathbf{M} + 1458.0\mathbf{S} \approx 2301.1\mathbf{M}$	
256	JQuartic	0**I**	17	$944.3\mathbf{M} + 1654.6\mathbf{S} \approx 2268.0\mathbf{M}$	large
256	JQuartic	1**I**	15	$935.4\mathbf{M} + 1661.6\mathbf{S} \approx 2264.6\mathbf{M}$	
256	JQuartic	2**I**	15	$928.4\mathbf{M} + 1661.6\mathbf{S} \approx 2257.7\mathbf{M}$	
256	JQuartic	3**I**	15	$926.3\mathbf{M} + 1661.6\mathbf{S} \approx 2255.6\mathbf{M}$	
256	Projective	0**I**	13	$1826.5\mathbf{M} + 1610.1\mathbf{S} \approx 3114.5\mathbf{M}$	large
256	Projective	1**I**	15	$1702.5\mathbf{M} + 1619.9\mathbf{S} \approx 2998.4\mathbf{M}$	≤ 116.1
256	Projective	2**I**	15	$1729.5\mathbf{M} + 1606.9\mathbf{S} \approx 3015.1\mathbf{M}$	
256	Projective	3**I**	15	$1724.5\mathbf{M} + 1608.9\mathbf{S} \approx 3011.6\mathbf{M}$	

TABLE 4.3. Optimal parameters for 512-bit scalars for each curve shape and each inversion strategy, assuming $\mathbf{S}/\mathbf{M} = 0.8$.

Bits	Curve shape	Inversions	m	Multiplications and squarings	$\mathbf{I}/\mathbf{M}$
512	2DIK	0**I**	29	$2033.5\mathbf{M} + 2964.9\mathbf{S} \approx 4405.4\mathbf{M}$	large
512	2DIK	1**I**	15	$1806.2\mathbf{M} + 2919.9\mathbf{S} \approx 4142.2\mathbf{M}$	≤ 263.2
512	2DIK	2**I**	25	$1779.1\mathbf{M} + 2903.7\mathbf{S} \approx 4102.1\mathbf{M}$	≤ 40.1
512	2DIK	3**I**	31	$1762.7\mathbf{M} + 2906.0\mathbf{S} \approx 4087.6\mathbf{M}$	≤ 14.5
512	3DIK	0**I**	29	$1867.5\mathbf{M} + 4054.2\mathbf{S} \approx 5110.9\mathbf{M}$	large
512	3DIK	1**I**	15	$1715.4\mathbf{M} + 3936.9\mathbf{S} \approx 4865.0\mathbf{M}$	≤ 245.9
512	3DIK	2**I**	21	$1700.1\mathbf{M} + 3915.1\mathbf{S} \approx 4832.2\mathbf{M}$	≤ 32.8
512	3DIK	3**I**	27	$1696.4\mathbf{M} + 3900.8\mathbf{S} \approx 4817.0\mathbf{M}$	≤ 15.2
512	Edwards	0**I**	29	$2394.6\mathbf{M} + 2113.5\mathbf{S} \approx 4085.4\mathbf{M}$	large
512	Edwards	1**I**	21	$2388.2\mathbf{M} + 2116.8\mathbf{S} \approx 4081.7\mathbf{M}$	
512	Edwards	2**I**	25	$2378.1\mathbf{M} + 2114.7\mathbf{S} \approx 4069.9\mathbf{M}$	
512	Edwards	3**I**	31	$2369.4\mathbf{M} + 2113.6\mathbf{S} \approx 4060.3\mathbf{M}$	
512	ExtJQuartic	0**I**	29	$2215.5\mathbf{M} + 2293.6\mathbf{S} \approx 4050.4\mathbf{M}$	large
512	ExtJQuartic	1**I**	19	$2204.0\mathbf{M} + 2313.5\mathbf{S} \approx 4054.8\mathbf{M}$	
512	ExtJQuartic	2**I**	21	$2196.2\mathbf{M} + 2314.3\mathbf{S} \approx 4047.7\mathbf{M}$	
512	ExtJQuartic	3**I**	23	$2194.4\mathbf{M} + 2323.3\mathbf{S} \approx 4053.1\mathbf{M}$	
512	Hessian	0**I**	29	$4583.4\mathbf{M} + 508.5\mathbf{S} \approx 4990.2\mathbf{M}$	large
512	Hessian	1**I**	19	$4510.4\mathbf{M} + 509.9\mathbf{S} \approx 4918.2\mathbf{M}$	≤ 72.0
512	Hessian	2**I**	25	$4490.3\mathbf{M} + 509.5\mathbf{S} \approx 4897.8\mathbf{M}$	≤ 20.4
512	Hessian	3**I**	31	$4488.6\mathbf{M} + 509.1\mathbf{S} \approx 4895.9\mathbf{M}$	
512	InvEdwards	0**I**	29	$2306.0\mathbf{M} + 2113.5\mathbf{S} \approx 3996.9\mathbf{M}$	large
512	InvEdwards	1**I**	21	$2299.0\mathbf{M} + 2116.2\mathbf{S} \approx 3992.0\mathbf{M}$	
512	InvEdwards	2**I**	25	$2292.0\mathbf{M} + 2113.1\mathbf{S} \approx 3982.5\mathbf{M}$	
512	InvEdwards	3**I**	27	$2296.5\mathbf{M} + 2106.3\mathbf{S} \approx 3981.6\mathbf{M}$	
512	JacIntersect	0**I**	31	$2490.8\mathbf{M} + 2202.2\mathbf{S} \approx 4252.6\mathbf{M}$	large
512	JacIntersect	1**I**	25	$2568.6\mathbf{M} + 2205.4\mathbf{S} \approx 4332.9\mathbf{M}$	
512	JacIntersect	2**I**	25	$2568.4\mathbf{M} + 2205.4\mathbf{S} \approx 4332.7\mathbf{M}$	
512	JacIntersect	3**I**	31	$2563.5\mathbf{M} + 2204.2\mathbf{S} \approx 4326.8\mathbf{M}$	
512	Jacobian	0**I**	29	$1362.0\mathbf{M} + 4404.0\mathbf{S} \approx 4885.2\mathbf{M}$	large
512	Jacobian	1**I**	31	$1151.2\mathbf{M} + 4384.0\mathbf{S} \approx 4658.4\mathbf{M}$	≤ 226.8
512	Jacobian	2**I**	21	$1194.5\mathbf{M} + 4420.7\mathbf{S} \approx 4731.1\mathbf{M}$	
512	Jacobian	3**I**	27	$1191.1\mathbf{M} + 4406.1\mathbf{S} \approx 4716.0\mathbf{M}$	
512	Jacobian-3	0**I**	29	$2373.0\mathbf{M} + 2887.5\mathbf{S} \approx 4683.0\mathbf{M}$	large
512	Jacobian-3	1**I**	31	$2161.3\mathbf{M} + 2868.8\mathbf{S} \approx 4456.3\mathbf{M}$	≤ 226.7
512	Jacobian-3	2**I**	21	$2205.7\mathbf{M} + 2904.0\mathbf{S} \approx 4528.9\mathbf{M}$	
512	Jacobian-3	3**I**	27	$2201.6\mathbf{M} + 2890.4\mathbf{S} \approx 4513.9\mathbf{M}$	
512	JQuartic	0**I**	29	$1799.4\mathbf{M} + 3302.6\mathbf{S} \approx 4441.5\mathbf{M}$	large
512	JQuartic	1**I**	21	$1789.5\mathbf{M} + 3319.5\mathbf{S} \approx 4445.1\mathbf{M}$	
512	JQuartic	2**I**	25	$1780.7\mathbf{M} + 3316.7\mathbf{S} \approx 4434.1\mathbf{M}$	
512	JQuartic	3**I**	31	$1772.4\mathbf{M} + 3310.9\mathbf{S} \approx 4421.1\mathbf{M}$	
512	Projective	0**I**	29	$3562.2\mathbf{M} + 3215.1\mathbf{S} \approx 6134.2\mathbf{M}$	large
512	Projective	1**I**	31	$3332.0\mathbf{M} + 3242.7\mathbf{S} \approx 5926.1\mathbf{M}$	≤ 208.1
512	Projective	2**I**	27	$3387.8\mathbf{M} + 3216.0\mathbf{S} \approx 5960.6\mathbf{M}$	
512	Projective	3**I**	31	$3369.9\mathbf{M} + 3215.7\mathbf{S} \approx 5942.4\mathbf{M}$	