From now on: non-binary field $k$; non-square $d \in k$.

$$E(k) = \{(x, y) \in k \times k : \quad x^2 + y^2 = 1 + dx^2y^2 \}$$

is a commutative group with

$$(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$$

defined by Edwards addition law:

$$x_3 = \frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2},$$

$$y_3 = \frac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2}.$$

Birationally equivalent to

$$(1/e)v^2 = u^3 + (4/e - 2)u^2 + u$$

where $e = 1 - d$. 
Represent \((x, y) \in E(k)\) by \((X : Y : Z) \in \mathbf{P}^2(k)\); i.e., \((X, Y, Z) \in k^3\) with \(Z \neq 0\) and \((X^2 + Y^2)Z^2 = Z^4 + dX^2Y^2\) represents \((X/Z, Y/Z) \in E(k)\).

\[\begin{align*}
10&M \quad \text{(10 field mults)} \\
+ &1S \quad \text{(1 field squaring)} \\
+ &1D \quad \text{(1 field mult by } d) \\
+ &7\text{add} \quad \text{(7 field additions)}
\end{align*}\]

to obtain sum \((X_3 : Y_3 : Z_3)\) of \((X_1 : Y_1 : Z_1), (X_2 : Y_2 : Z_2)\).

Don’t have to make distinctions for equal inputs, negatives, etc.
What if we want to make distinctions to gain speed? For example, speed up doubling?

\[ 2(x, y) = \begin{pmatrix} \frac{xy + yx}{1 + dxxyy} & \frac{yy - xx}{1 - dxxyy} \end{pmatrix} \]
What if we want to make distinctions to gain speed? For example, speed up doubling?

\[
2(x, y) = \begin{pmatrix}
\frac{xy + yx}{1 + dxxyy} & \frac{yy - xx}{1 - dxxyy}
\end{pmatrix}
\begin{pmatrix}
\frac{2xy}{1 + dx^2y^2} & \frac{y^2 - x^2}{1 - dx^2y^2}
\end{pmatrix}
\] 
save mults!
What if we want to make distinctions to gain speed? For example, speed up doubling?

\[ 2(x, y) = \left( \begin{array}{cc}
\frac{xy + yx}{1 + dxxyy'} & \frac{yy - xx}{1 - dxxyyy} \\
\frac{2xy}{1 + dx^2y^2'} & \frac{y^2 - x^2}{1 - dx^2y^2}
\end{array} \right) \]

save mults!

\[ = \left( \begin{array}{cc}
\frac{2xy}{x^2 + y^2'} & \frac{y^2 - x^2}{1 - dx^2y^2}
\end{array} \right) \]

low deg (Joye)
What if we want to make distinctions to gain speed?
For example, speed up doubling?

\[ 2(x, y) \]

\[ = \left( \begin{array}{cc} xy + yx & yy - xx \\ 1 + dxxyy' & 1 - dxxyyy \\ \end{array} \right) \]

\[ = \left( \begin{array}{cc} 2xy & y^2 - x^2 \\ 1 + dx^2y^2' & 1 - dx^2y^2 \\ \end{array} \right) \]

\[ = \left( \begin{array}{cc} 2xy & y^2 - x^2 \\ x^2 + y^2' & 1 - dx^2y^2 \\ \end{array} \right) \]

\[ = \left( \begin{array}{cc} 2xy & y^2 - x^2 \\ x^2 + y^2' & 2 - x^2 - y^2 \\ \end{array} \right) \]

save mults!
low deg (Joye)
even lower
What if we want to make distinctions to gain speed?
For example, speed up doubling?

\[2(x, y)\]

\[
= \left(\begin{array}{cc}
\frac{xy + yx}{1 + dxxyy'} & \frac{yy - xx}{1 - dxxyyy'} \\
\frac{2xy}{1 + dx^2y^2'} & \frac{y^2 - x^2}{1 - dx^2y^2'}
\end{array}\right)
\]

save mults!

\[
= \left(\begin{array}{cc}
\frac{2xy}{x^2 + y^2'} & \frac{y^2 - x^2}{1 - dx^2y^2'} \\
\frac{2xy}{x^2 + y^2'} & \frac{y^2 - x^2}{2 - x^2 - y^2}
\end{array}\right)
\]

low deg (Joye)

even lower

\[
= \left(\begin{array}{cc}
\frac{(x + y)^2}{x^2 + y^2} - 1, & \frac{y^2 - x^2}{2 - x^2 - y^2}
\end{array}\right)
\]
3M  (3 field mults)
+ 4S  (4 field squarings)
+ 6add  (6 field additions)
to double \((X_1 : Y_1 : Z_1)\):

\[
B = (X_1 + Y_1)^2, \\
C = X_1^2, \\
D = Y_1^2, \\
E = C + D, \\
H = Z_1^2, \\
J = E - 2H, \\
X_3 = (B - E)J, \\
Y_3 = E(C - D), \\
Z_3 = EJ.
\]
Comparison of doubling costs if curve parameters are small:

<table>
<thead>
<tr>
<th>System</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Projective</td>
<td>$5M + 6S$</td>
</tr>
<tr>
<td>Projective if $a = -3$</td>
<td>$7M + 3S$</td>
</tr>
<tr>
<td>Hessian</td>
<td>$7M + 1S$</td>
</tr>
<tr>
<td>Doche/Icart/Kohel 3</td>
<td>$2M + 7S$</td>
</tr>
<tr>
<td>Jacobian</td>
<td>$1M + 8S$</td>
</tr>
<tr>
<td>Jacobian if $a = -3$</td>
<td>$3M + 5S$</td>
</tr>
<tr>
<td>Jacobi quartic</td>
<td>$2M + 6S$</td>
</tr>
<tr>
<td>Jacobi intersection</td>
<td>$3M + 4S$</td>
</tr>
<tr>
<td>Edwards</td>
<td>$3M + 4S$</td>
</tr>
<tr>
<td>Doche/Icart/Kohel 2</td>
<td>$2M + 5S$</td>
</tr>
</tbody>
</table>

EFD! EFD! EFD! EFD! EFD! EFD!

e.g. Doche/Icart/Kohel paper says $3M + 4S$ for Doche/Icart/Kohel 2.
Jacobian $a = -3$ vs. Edwards:

<table>
<thead>
<tr>
<th></th>
<th>Jac-3</th>
<th>Edwards</th>
</tr>
</thead>
<tbody>
<tr>
<td>Double</td>
<td>$3M + 5S$</td>
<td>$3M + 4S$</td>
</tr>
<tr>
<td>Triple</td>
<td>$7M + 7S$</td>
<td>$9M + 4S$</td>
</tr>
<tr>
<td>Add</td>
<td>$11M + 5S$</td>
<td>$10M + 1S + 1D$</td>
</tr>
<tr>
<td>Readd</td>
<td>$10M + 4S$</td>
<td>$10M + 1S + 1D$</td>
</tr>
<tr>
<td>Mixed</td>
<td>$7M + 4S$</td>
<td>$9M + 1S + 1D$</td>
</tr>
<tr>
<td>Unified</td>
<td>unclear</td>
<td>$10M + 1S + 1D$</td>
</tr>
</tbody>
</table>

Jac-3 speedup for readd:
Chudnovsky/Chudnovsky 1986;
“Chudnovsky coordinates” etc.

Edwards tripling:
Bernstein/Birkner/Lange/Peters
2007; independently
A sensible ElGamal-type system (van Duin, sci.crypt, 2006):

Everyone knows standard point $B$, prime order $q$, on “Curve25519”: $\mathbb{Z}/(2^{255} - 19); d = 1 - 1/121666$.

Signer has 32-byte secret key $n$. Everyone knows signer’s 32-byte public key: compressed $nB$.

To verify $(m, \text{compressed } R, t)$: verify $tB = H(R, m)R + nB$.

To sign $m$: generate a secret $s$; $R = sB; t = H(R, m)s + n \mod q$.

Notes: 1. No inversions mod $q$.
2. Send $R$, not $H(R, m)$. 
Batch verification of many $t_i B - h_i R_i = S_i$: check
$$\sum_i u_i t_i B - \sum_i u_i h_i R_i - \sum_i u_i S_i = 0$$
for random 128-bit $u_i$.
(Naccache et al., Eurocrypt 1994; Bellare et al., Eurocrypt 1998)

Use subtractive multi-scalar multiplication algorithm:
if $n_1 \geq n_2 \geq \cdots$ then
$$n_1 P_1 + n_2 P_2 + n_3 P_3 + \cdots = (n_1 - qn_2) P_1 + n_2(qP_1 + P_2) + n_3 P_3 + \cdots$$
where $q = \lfloor n_1/n_2 \rfloor$.
(credited to Bos and Coster by de Rooij, Eurocrypt 1994; see also tweaks by Wei Dai, 2007)
Verifying 100 signatures requires a 201-scalar mult with 101 256-bit scalars and 100 128-bit scalars.

Subtractive algorithm then uses \( \approx 24.4 \cdot 256 \) readds and \( \approx 0.8 \cdot 256 \) mixed adds.

\( S/M = 0.8 \), small parameters:
\( \approx 845M/\text{signature with Jacobian}; \)
\( \approx 695M/\text{signature with Edwards}. \)

**Use Edwards coordinates!**

Can similar speeds be achieved by genus-2 hyperelliptic curves? Current attempts seem very slow.
We’ve counted mults (with various S/M, D/M) for Edwards, Jac-3, Hessian, et al. in NAF; width-4 sliding windows; JSF; accelerated ECDSA; batch verification, as above; fixed-point scalar mult; and several side-channel situations.


Need to measure overheads too. Planning new Edwards software. Expect new speed records.


Importance of doubling/tripling ratio (200 bits, all shapes)
New directions in ECC

We’re working on several items:

cr.yp.to/newelliptic.html
New directions in ECC

We’re working on several items:
Edwards for precomputation!

cr.yp.to/newelliptic.html
New directions in ECC

We’re working on several items:
Edwards for precomputation!
Edwards for pairings!

cr.yp.to/newelliptic.html
New directions in ECC

We’re working on several items:
Edwards for precomputation!
Edwards for pairings!
Edwards for president!

cr.yp.to/newelliptic.html
New directions in ECC

We’re working on several items:

Edwards for precomputation!
Edwards for pairings!
Edwards for president!
Edwards implementations!

cr.yp.to/newelliptic.html
New directions in ECC

We’re working on several items:

Edwards for precomputation!
Edwards for pairings!
Edwards for president!
Edwards implementations!
Edwards standardization!

cr.yp.to/newelliptic.html
New directions in ECC

We’re working on several items:

Edwards for precomputation!
Edwards for pairings!
Edwards for president!
Edwards implementations!
Edwards standardization!

And beyond ECC:
Edwards for ECM!

cr.yp.to/newelliptic.html
New directions in ECC

We’re working on several items:

Edwards for precomputation!
Edwards for pairings!
Edwards for president!
Edwards implementations!
Edwards standardization!

And beyond ECC:
Edwards for ECM!
Edwards for ECPP!

cr.yp.to/newelliptic.html
New directions in ECC

We’re working on several items:

Edwards for precomputation!
Edwards for pairings!
Edwards for president!
Edwards implementations!
Edwards standardization!

And beyond ECC:
Edwards for ECM!
Edwards for ECPP!
Edwards for ECXYZ!

cr.yp.to/newelliptic.html
New directions in ECC

We’re working on several items:

Edwards for precomputation!
Edwards for pairings!
Edwards for president!
Edwards implementations!
Edwards standardization!

And beyond ECC:
Edwards for ECM!
Edwards for ECPP!
Edwards for ECXYZ!
Return of the Hyperelliptic!

cr.yp.to/newelliptic.html