Twisted Hessian curves

cr.yp.to/papers.html#hessian

Daniel J. Bernstein
University of Illinois at Chicago &
Technische Universiteit Eindhoven

Joint work with:

Chitchanok Chuengsatiansup
Technische Universiteit Eindhoven

David Kohel
Aix-Marseille Université

Tanja Lange
Technische Universiteit Eindhoven
1986 Chudnovsky–Chudnovsky, “Sequences of numbers generated by addition in formal groups and new primality and factorization tests”:

“The crucial problem becomes the choice of the model of an algebraic group variety, where computations mod $p$ are the least time consuming.”

Most important computations: ADD is $P, Q \mapsto P + Q$. DBL is $P \mapsto 2P$. 
“It is preferable to use models of elliptic curves lying in low-dimensional spaces, for otherwise the number of coordinates and operations is increasing. This limits us . . . to 4 basic models of elliptic curves.”

Short Weierstrass:
\[ y^2 = x^3 + ax + b. \]

Jacobi intersection:
\[ s^2 + c^2 = 1, \quad as^2 + d^2 = 1. \]

Jacobi quartic: \[ y^2 = x^4 + 2ax^2 + 1. \]

Hessian: \[ x^3 + y^3 + 1 = 3dxy. \]
“Our experience shows that the expression of the law of addition on the cubic Hessian form (d) of an elliptic curve is by far the best and the prettiest.”

\[
X_3 = Y_1 X_2 \cdot Y_1 Z_2 - Z_1 Y_2 \cdot X_1 Y_2, \\
Y_3 = X_1 Z_2 \cdot X_1 Y_2 - Y_1 X_2 \cdot Z_1 X_2, \\
Z_3 = Z_1 Y_2 \cdot Z_1 X_2 - X_1 Z_2 \cdot Y_1 Z_2.
\]

12M for ADD, where M is the cost of multiplication in the field.

8.4M for DBL, assuming 0.8M for the cost of squaring in the field.
1990s: ECC standards instead use short Weierstrass curves in Jacobian coordinates for “the fastest arithmetic”.

15.2\text{M} for ADD, much slower than Hessian.

Why is this a good idea?
1990s: ECC standards instead use short Weierstrass curves in Jacobian coordinates for “the fastest arithmetic”.

15.2\text{M} for ADD, much slower than Hessian.

Why is this a good idea?
Answer: Only 7.2\text{M} for DBL with Chudnovsky–Chudnovsky formula.
1990s: ECC standards instead use short Weierstrass curves in Jacobian coordinates for “the fastest arithmetic”.

15.2M for ADD, much slower than Hessian.

Why is this a good idea?
Answer: Only 7.2M for DBL with Chudnovsky–Chudnovsky formula.

2001 Bernstein: 15M, 7M.
1990s: ECC standards instead use short Weierstrass curves in Jacobian coordinates for “the fastest arithmetic”.

15.2M for ADD, much slower than Hessian.

Why is this a good idea? Answer: Only 7.2M for DBL with Chudnovsky–Chudnovsky formula.

2001 Bernstein: 15M, 7M.

Compared to Hessian, Weierstrass saves 4M in typical DBL-DBL-DBL-DBL-DBL-ADD.
2007 Bernstein–Lange: generalize, analyze speed, completeness.

Example: \( x^2 + y^2 = 1 - 30x^2y^2 \).
Sum of \((x_1, y_1)\) and \((x_2, y_2)\) is
\[ \left( \frac{x_1y_2+y_1x_2}{1-30x_1x_2y_1y_2}, \frac{y_1y_2-x_1x_2}{1+30x_1x_2y_1y_2} \right). \]
2007 Bernstein–Lange:
10.8\text{M} for ADD, 6.2\text{M} for DBL.
2007 Bernstein–Lange: 10.8M for ADD, 6.2M for DBL.

2008 Hisil–Wong–Carter–Dawson: just 8M for ADD.
2007 Bernstein–Lange:
10.8M for ADD, 6.2M for DBL.

2008 Hisil–Wong–Carter–Dawson:
just 8M for ADD.
\( y^2 = x^3 - 0.4x + 0.7 \)
The Weierstrass-turtle: old, trusted and slow. Warning: (picture) incomplete!
\[ x^2 + y^2 = 1 - 300x^2y^2 \]
The Edwards starfish: new, fast and complete!
\[ x^2 = y^4 - 1.9y^2 + 1 \]
The Jacobi-quartic squid: can be extended to XXYZZR giant squid.
$x^3 - y^3 + 1 = 0.3xy$
The Hessian-ray: uniform

but

not strongly so
Mar
Zoom
Faster Hessian arithmetic

2007 Hisil–Carter–Dawson: 7.8M for DBL.
Faster Hessian arithmetic

2007 Hisil–Carter–Dawson: $7.8M$ for DBL.

2010 Hisil: $11M$ for ADD.
Faster Hessian arithmetic

2007 Hisil–Carter–Dawson: 7.8M for DBL.

2010 Hisil: 11M for ADD.

Hessian tied with Weierstrass for DBL-DBL-DBL-DBL-DBL-ADD.

Need to zoom in closer: analyze exact $S/M$, overhead for checking for special cases, extra DBL, extra ADD, etc.
Faster Hessian arithmetic

2007 Hisil–Carter–Dawson: 7.8M for DBL.

2010 Hisil: 11M for ADD.

Hessian tied with Weierstrass for DBL-DBL-DBL-DBL-DBL-ADD.

Need to zoom in closer: analyze exact S/M, overhead for checking for special cases, extra DBL, extra ADD, etc.

Or speed up Hessian more.
Faster Hessian arithmetic

2007 Hisil–Carter–Dawson: \(7.8M\) for DBL.

2010 Hisil: \(11M\) for ADD.

Hessian tied with Weierstrass for DBL-DBL-DBL-DBL-DBL-ADD.

Need to zoom in closer:
analyze exact \(S/M\), overhead for checking for special cases, extra DBL, extra ADD, etc.

Or speed up Hessian more.

New: \(7.6M\) for DBL.
New (announced July 2009):

Generalize to more curves:

**twisted Hessian curves**

\[ aX^3 + Y^3 + Z^3 = dXYZ \]

with \( a(27a - d^3) \neq 0 \).

2007 7.8M DBL idea fails, but
2010 11M ADD generalizes,
new 7.6M DBL generalizes.
New (announced July 2009):

Generalize to more curves:

**twisted Hessian curves**

\[ aX^3 + Y^3 + Z^3 = dXYZ \]

with \( a(27a - d^3) \neq 0 \).

2007 7.8M DBL idea fails, but 2010 11M ADD generalizes, new 7.6M DBL generalizes.

**Rotate** addition law
so that it also works for DBL;
**complete** if \( a \) is not a cube.
Eliminates special-case overhead,
helps stop side-channel attacks.
Triplings (assuming $d \neq 0$)

TPL is $P \mapsto 3P$.

2007 Hisil–Carter–Dawson:

12.8M for Hessian TPL.

Generalizes to twisted Hessian.
Triplings (assuming $d \neq 0$)

TPL is $P \mapsto 3P$.

2007 Hisil–Carter–Dawson:
12.8M for Hessian TPL.

Generalizes to twisted Hessian.

2015 Kohel: 11.2M.
Triplings (assuming $d \neq 0$)

TPL is $P \mapsto 3P$.

2007 Hisil–Carter–Dawson: 12.8M for Hessian TPL.

Generalizes to twisted Hessian.

2015 Kohel: 11.2M.

New: 10.8M assuming field with fast primitive $\sqrt[3]{1}$; e.g., $F_q[\omega]/(\omega^2 + \omega + 1)$, or $F_p$ with $7p = 2^{298} + 2^{149} + 1$.

(More history in small char. See paper for details.)
If $aX^3 + Y^3 + Z^3 = dXYZ$
then $VW(V + dU + aW) = U^3$
where 
\[ U = -XYZ, \ V = Y^3, \ W = X^3. \]

If $VW(V + dU + aW) = U^3$
then $aX_3^3 + Y_3^3 + Z_3^3 = dX_3Y_3Z_3$
where $Q = dU$, $R = aW$, 
\[ S = -(V + Q + R), \]
\[ dX_3 = R^3 + S^3 + V^3 - 3RSV, \]
\[ Y_3 = RS^2 + SV^2 + VR^2 - 3RSV, \]
\[ Z_3 = RV^2 + SR^2 + VS^2 - 3RSV. \]

Compose these 3-isogenies:
\[ (X_3 : Y_3 : Z_3) = 3(X : Y : Z). \]
To quickly triple \((X : Y : Z)\):

Three cubings for \(R, S, V\).

For three choices of constants \((\alpha, \beta, \gamma)\) compute

\[(\alpha R + \beta S + \gamma V) \cdot (\alpha S + \beta V + \gamma R) \cdot (\alpha V + \beta R + \gamma S) = \alpha \beta \gamma dX_3 + (\alpha \beta^2 + \beta \gamma^2 + \gamma \alpha^2)Y_3 + (\beta \alpha^2 + \gamma \beta^2 + \alpha \gamma^2)Z_3 + (\alpha + \beta + \gamma)^3 RSV.

Also use \(a(R + S + V)^3 = d^3 RSV\). Solve for \(dX_3, Y_3, Z_3\).
2015 Kohel’s 11.2\textbf{M}
(4 cubings + 4 mults)
introduced this TPL idea with
\((\alpha, \beta, \gamma) = (1, 1, 1),\)
\((\alpha, \beta, \gamma) = (1, -1, 0),\)
\((\alpha, \beta, \gamma) = (1, 1, 0).\)
2015 Kohel’s 11.2M
(4 cubings + 4 mults)
introduced this TPL idea with
\((\alpha, \beta, \gamma) = (1, 1, 1)\),
\((\alpha, \beta, \gamma) = (1, -1, 0)\),
\((\alpha, \beta, \gamma) = (1, 1, 0)\).

New 10.8M (6 cubings)
makes faster choices
assuming fast primitive \(\omega = \sqrt[3]{1}\):
\((\alpha, \beta, \gamma) = (1, 1, 1)\),
\((\alpha, \beta, \gamma) = (1, \omega, \omega^2)\),
\((\alpha, \beta, \gamma) = (1, \omega^2, \omega)\).
Are triplings useful?

2005 Dimitrov–Imbert–Mishra
“double-base chains”: e.g.,
compute $314159P$ as
$2^{15}3^2P + 2^{11}3^2P + 2^83^1P$
$+ 2^43^1P - 2^03^0P$.
2TPL, 15DBL, 4ADD.

2006 Doche–Imbert
generalized double-base chains:
e.g., compute $314159P$ as
$2^{12}3^33P - 2^73^5P - 2^43^17P - 2^03^0P$
after precomputing $3P, 5P, 7P$.
3TPL, 13DBL, 6ADD.
Not good for constant time. Good for signature verification, factorization, math, etc.

Also need time to compute chain. Good for scalars used many times.
Not good for constant time. Good for signature verification, factorization, math, etc.

Also need time to compute chain. Good for scalars used many times.

Analysis+optimization from 2007 Bernstein–Birkner–Lange–Peters:

Double-base chains speed up Weierstrass curves slightly:
9.29M/bit for 256-bit scalars.

Revisit conclusions using latest Hessian formulas, latest double-base techniques.
Revisit conclusions using latest Hessian formulas, latest double-base techniques.

New: $8.77 \text{M}$/bit for 256 bits.
Revisit conclusions using latest Hessian formulas, latest double-base techniques.

New: $8.77\times 10^7$/bit for 256 bits.

Comparison to Weierstrass for 1-bit, 2-bit, \ldots, 64-bit scalars:

Uses 2008 Doche–Habsieger “tree search” and some new improvements: e.g., account for costs of ADD, DBL, TPL.
Summary:
Twisted Hessian curves solidly beat Weierstrass.

Chuengsatiansup talk tomorrow: even better double-base chains from shortest paths in DAG—and also new Edwards speeds!