ECC minicourse

Daniel J. Bernstein
University of Illinois at Chicago

Tanja Lange
Technische Universiteit Eindhoven

January 2010 news:
An academic team announces successful RSA-768 factorization.
Used $\approx$ 2 years of computation on $\approx$ 1000 CPU cores.
"Factoring a 1024-bit RSA modulus would be about a thousand times harder."

January 2010 news:
An academic team announces successful RSA-768 factorization.
Used $\approx$ 2 years of computation on $\approx$ 1000 CPU cores.
"Factoring a 1024-bit RSA modulus would be about a thousand times harder."

Many users of 1024-bit RSA:
https://www.fnb.co.za,
the root DNSSEC trial, etc.
2009 Kolkman et al.: "It is estimated that most zones can safely use 1024-bit keys for at least the next ten years."

1000 cores in perspective:

Typical laptop has 2 cores.

1000 cores in perspective:

Typical laptop has 2 cores.

A GTX 295 graphics card
has 60 cores (“MPs”).

1000 cores in perspective:

Typical laptop has 2 cores.

A GTX 295 graphics card has 60 cores ("MPs").

EPFL's 200-Playstation cluster has 1200 cores.

1000 cores in perspective:

Typical laptop has 2 cores.

A GTX 295 graphics card has 60 cores ("MPs").

EPFL's 200-Playstation cluster has 1200 cores.

Dan has an account on the TACC Ranger supercomputer, which has 62976 cores.

1000 cores in perspective:

Typical laptop has 2 cores.

A GTX 295 graphics card has 60 cores ("MPs").

EPFL's 200-Playstation cluster has 1200 cores.

Dan has an account on the TACC Ranger supercomputer, which has 62976 cores.

The Conficker/Downadup criminal-controlled botnet has $\approx 10\,000\,000$ cores.

2003 Shamir et al.:
An attacker building ASICs
for 10 million USD can break
RSA-1024 in a year.

2003 RSA company:
Move to 2048 bits “over the
remainder of this decade.”

2003 Shamir et al.:
An attacker building ASICs
for 10 million USD can break
RSA-1024 in a year.

2003 RSA company:
Move to 2048 bits “over the
remainder of this decade.”

2007 NIST: Same.

2003 Shamir et al.:
An attacker building ASICs
for 10 million USD can break
RSA-1024 in a year.

2003 RSA company:
Move to 2048 bits "over the
remainder of this decade."

2007 NIST: Same.

Another big reason to worry:
Attackers with more money
can use *batch* algorithms
that save time in breaking
many keys together.

A 1024-bit RSA key is built from two secret 512-bit primes.

There are $\approx 2^{503}$
possible 512-bit primes.
Can't imagine trying them all.
But the attacks are much faster:
only $\approx 2^{80}$ calculations.

A 1024-bit RSA key is built from two secret 512-bit primes.

There are $\approx 2^{503}$
possible 512-bit primes.
Can't imagine trying them all.
But the attacks are much faster:
only $\approx 2^{80}$ calculations.

2048-bit key: 1024-bit primes;
$\approx 2^{1014}$ possible primes.
Still below modern standards!
Attacks: $\approx 2^{112}$ calculations.

A 1024-bit RSA key is built from two secret 512-bit primes.

There are $\approx 2^{503}$
possible 512-bit primes.
Can't imagine trying them all.
But the attacks are much faster:
only $\approx 2^{80}$ calculations.

2048-bit key: 1024-bit primes;
$\approx 2^{1014}$ possible primes.
Still below modern standards!
Attacks: $\approx 2^{112}$ calculations.

3072-bit key: 1536-bit primes;
$\approx 2^{1526}$ possible primes.
Attacks: $\approx 2^{128}$ calculations.

Attacks use "index calculus"
= "combining congruences."
Long history, including
many major improvements:
1975, CFRAC;
1977, linear sieve (LS);
1982, quadratic sieve (QS);
1990, number-field sieve (NFS).
Also many smaller improvements.

Costs of these algorithms for
breaking RSA-1024, RSA-2048:
$\approx 2^{120}$, 2^{170}, CFRAC;
$\approx 2^{110}$, 2^{160}, LS;
$\approx 2^{100}$, 2^{150}, QS;
$\approx 2^{80}$, 2^{112}, NFS.

1977: RSA is introduced.

1985: Miller proposes switching
from RSA to elliptic curves.
Explains several obstacles
to congruence-combination
attacks on elliptic curves.

Subsequent ECC history:
Negligible security losses.

Subsequent RSA history:
Continued security losses
from improved algorithms
for combining congruences.
Major loss in 1990 (NFS);
many smaller losses since then.

256-bit ECC keys match
security of 3072-bit RSA keys.

When properly implemented,
256-bit ECC is much faster
than 3072-bit RSA for
almost all real-world applications.

ANSI, IEEE, NIST issued
ECC standards ten years ago.
US government "Suite B"
now prohibits RSA, requires ECC.

For much more information see
the Handbook of Elliptic and
Hyperelliptic Curve Cryptography:
www.hyperelliptic.org/HEHCC

Diffie-Hellman key exchange

Uses public base, e.g. $g = 2$,
and prime, e.g. $p = 11$.

User A picks random
secret integer a, e.g. $a = 4$,
and computes $h_A = g^a \bmod p$,
e.g. $h_A = 2^4 = 16 \equiv 5 \bmod 11$.

User B picks random
secret integer b, e.g. $b = 3$,
and computes $h_B = g^b \bmod p$,
e.g. $h_B = 2^3 = 8 \equiv 8 \bmod 11$.

Then A sends h_A to B
and B sends h_B to A.

Finally A computes ${h_B}^a \bmod p$,
e.g. for $p = 11$:
$8^4 = (8^2)^2 = 64^2 \equiv (-2)^2 \equiv 4$;
and B computes ${h_A}^b \bmod p$,
e.g. for $p = 11$:
$5^3 = (25) \cdot 5 \equiv 3 \cdot 5 \equiv 4$.

Both results are the same.

No surprise since
$h_B^a = (g^b)^a = g^{ab} = (g^a)^b = h_A^b$.

If a and b are secret then so
is g^{ab}; value can be used in
symmetric crypto.

Problems

The prime $p = 11$ is too small: attacker can read off a or b and then imitate A or B.
Solution: use much larger primes.

The exponent b is too small: $8 = 2^3$ over the integers.
Solution: big a, b so g^a, g^b are much larger than p.
This happens automatically for random a and b and large p.

Biggest problem: Index calculus.

An adapted version
of the index calculus method
works for any prime p.

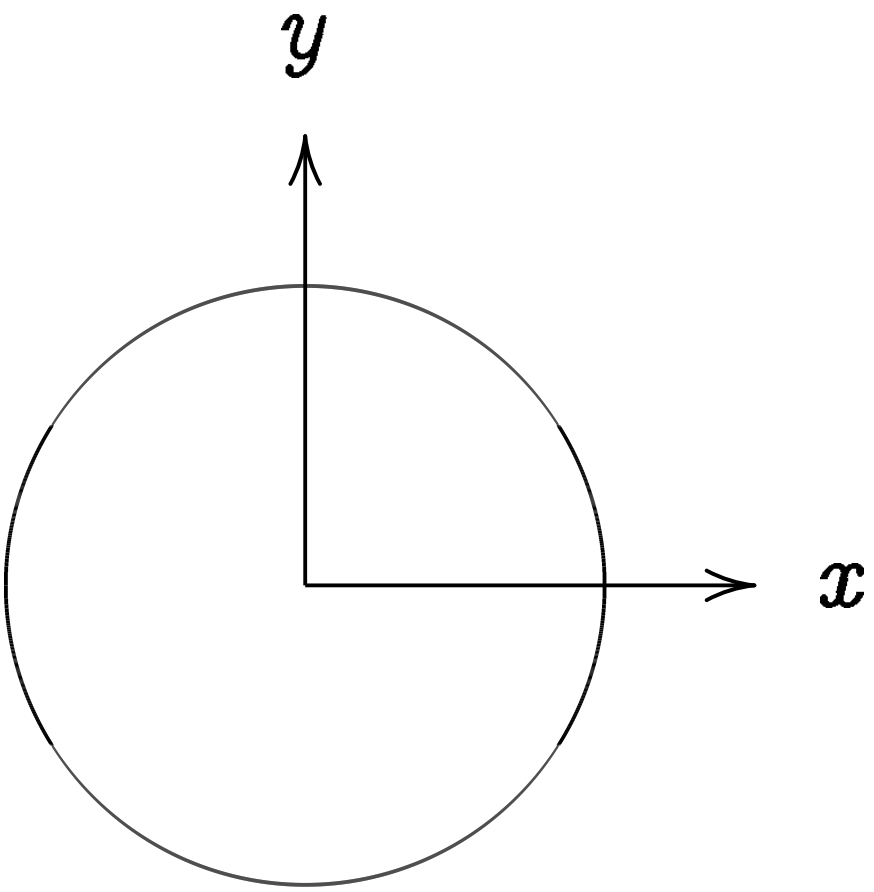
Also works for the generalization
to finite fields $\mathbf{F}_q$ with $q = p^r$.

Solution: huge primes $q \approx N$,
where N is an RSA modulus.

No benefit compared to RSA?!

Better solution: Alice and Bob
did not actually use any property
other than g is a generator of
a cyclic group. There are many
cyclic groups.

The clock



This is the curve $x^2 + y^2 = 1$.

Warning:

This is *not* an elliptic curve.

"Elliptic curve" $\neq$ "ellipse."

Examples of points on this curve:

Examples of points on this curve:
$(0, 1) =$ “12:00”.

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

$(-1, 0) =$ “9:00”.

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

$(-1, 0) =$ “9:00”.

$(\sqrt{3/4}, 1/2) =$

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

$(-1, 0) =$ “9:00”.

$(\sqrt{3/4}, 1/2) =$ “2:00”.

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

$(-1, 0) =$ “9:00”.

$(\sqrt{3/4}, 1/2) =$ “2:00”.

$(1/2, -\sqrt{3/4}) =$

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

$(-1, 0) =$ “9:00”.

$(\sqrt{3/4}, 1/2) =$ “2:00”.

$(1/2, -\sqrt{3/4}) =$ “5:00”.

$(-1/2, -\sqrt{3/4}) =$

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

$(-1, 0) =$ “9:00”.

$(\sqrt{3/4}, 1/2) =$ “2:00”.

$(1/2, -\sqrt{3/4}) =$ “5:00”.

$(-1/2, -\sqrt{3/4}) =$ “7:00”.

Examples of points on this curve:

$(0, 1) =$ "12:00".

$(0, -1) =$ "6:00".

$(1, 0) =$ "3:00".

$(-1, 0) =$ "9:00".

$(\sqrt{3/4}, 1/2) =$ "2:00".

$(1/2, -\sqrt{3/4}) =$ "5:00".

$(-1/2, -\sqrt{3/4}) =$ "7:00".

$(\sqrt{1/2}, \sqrt{1/2}) =$ "1:30".

$(3/5, 4/5)$. $(-3/5, 4/5)$.

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

$(-1, 0) =$ “9:00”.

$(\sqrt{3/4}, 1/2) =$ “2:00”.

$(1/2, -\sqrt{3/4}) =$ “5:00”.

$(-1/2, -\sqrt{3/4}) =$ “7:00”.

$(\sqrt{1/2}, \sqrt{1/2}) =$ “1:30”.

$(3/5, 4/5)$. $(-3/5, 4/5)$.

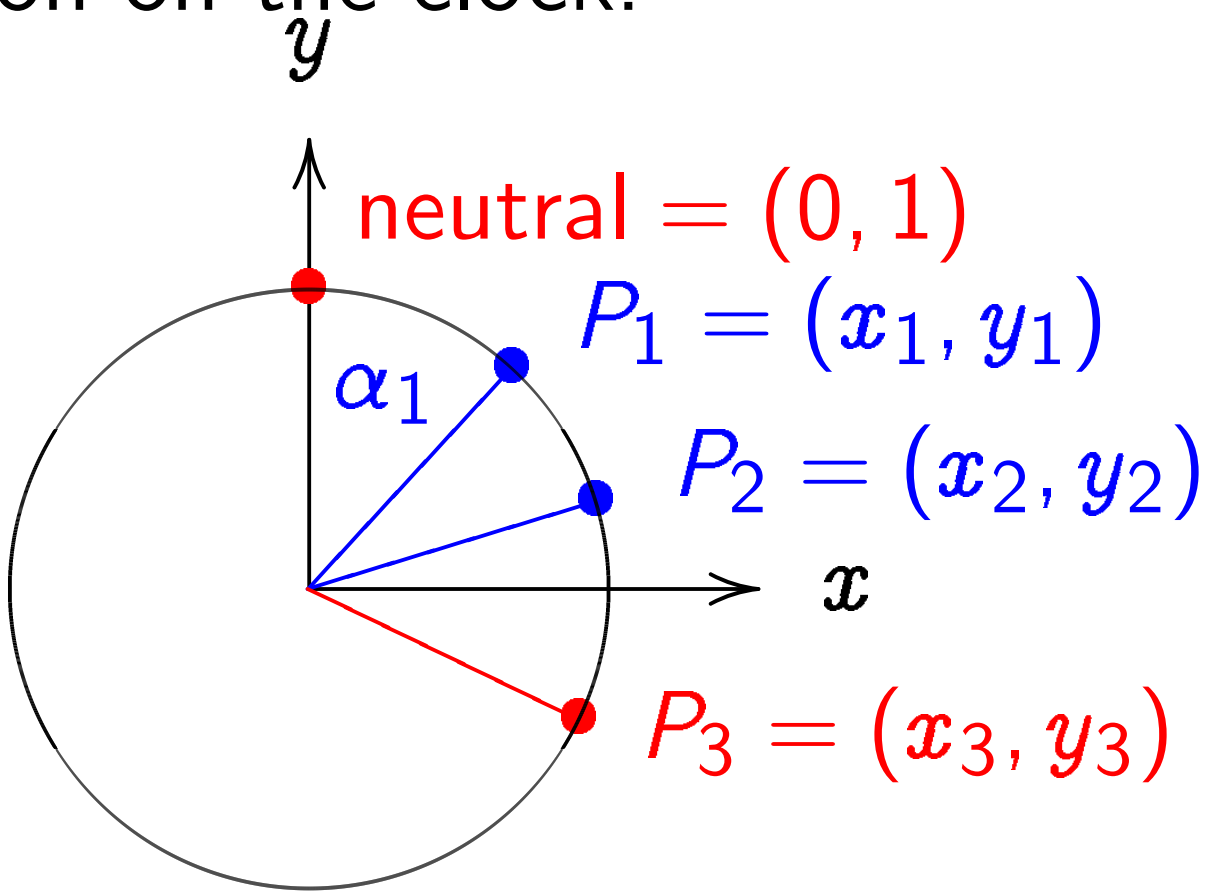
$(3/5, -4/5)$. $(-3/5, -4/5)$.

$(4/5, 3/5)$. $(-4/5, 3/5)$.

$(4/5, -3/5)$. $(-4/5, -3/5)$.

Many more.

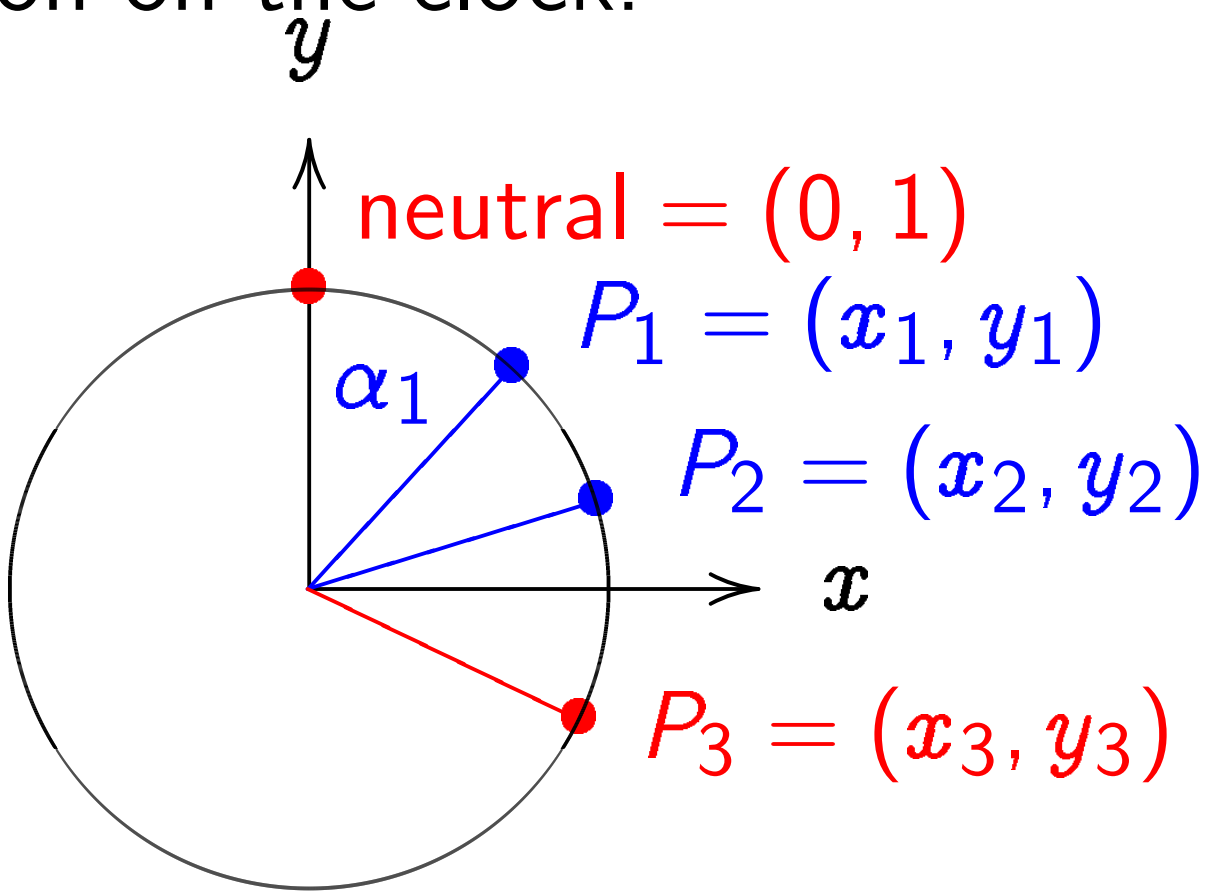
Addition on the clock:



$x^2 + y^2 = 1$, parametrized by

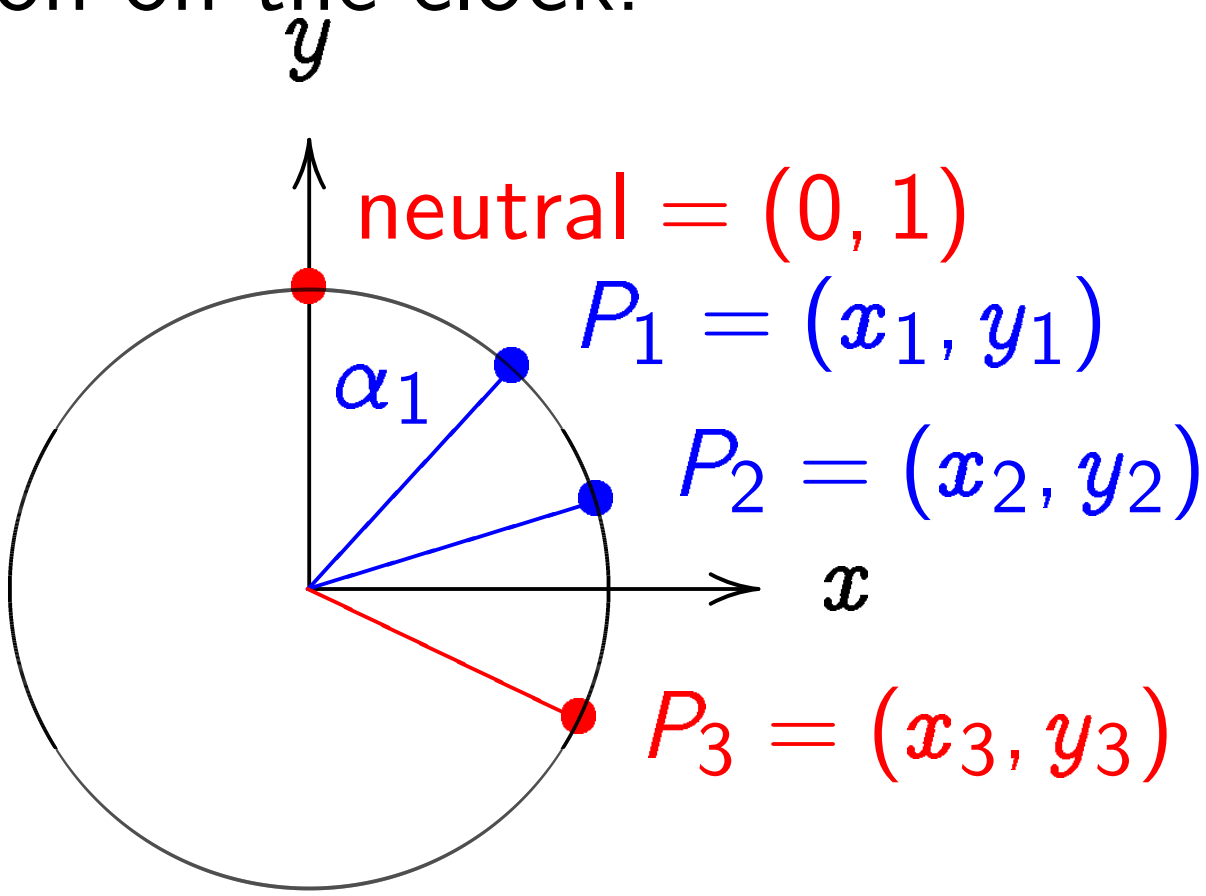
$x = \sin \alpha, \quad y = \cos \alpha.$

Addition on the clock:



$x^2 + y^2 = 1$, parametrized by $x = \sin\alpha, \quad y = \cos\alpha$. Recall $(\sin(\alpha_1 + \alpha_2), \cos(\alpha_1 + \alpha_2)) =$

Addition on the clock:

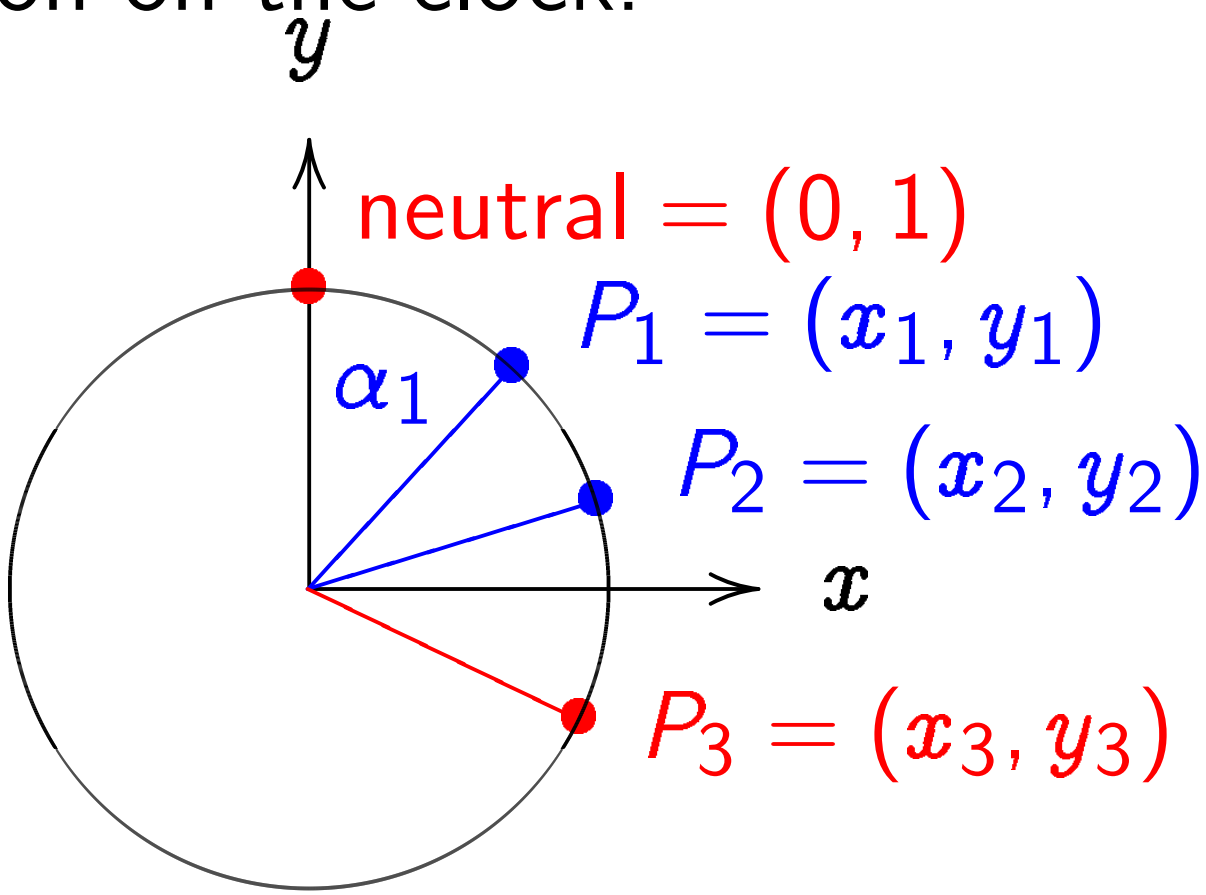


$x^2 + y^2 = 1$, parametrized by $x = \sin\alpha, \quad y = \cos\alpha$. Recall

$(\sin(\alpha_1 + \alpha_2), \cos(\alpha_1 + \alpha_2)) =$

$(\sin\alpha_1 \cos\alpha_2 + \cos\alpha_1 \sin\alpha_2,$

Addition on the clock:



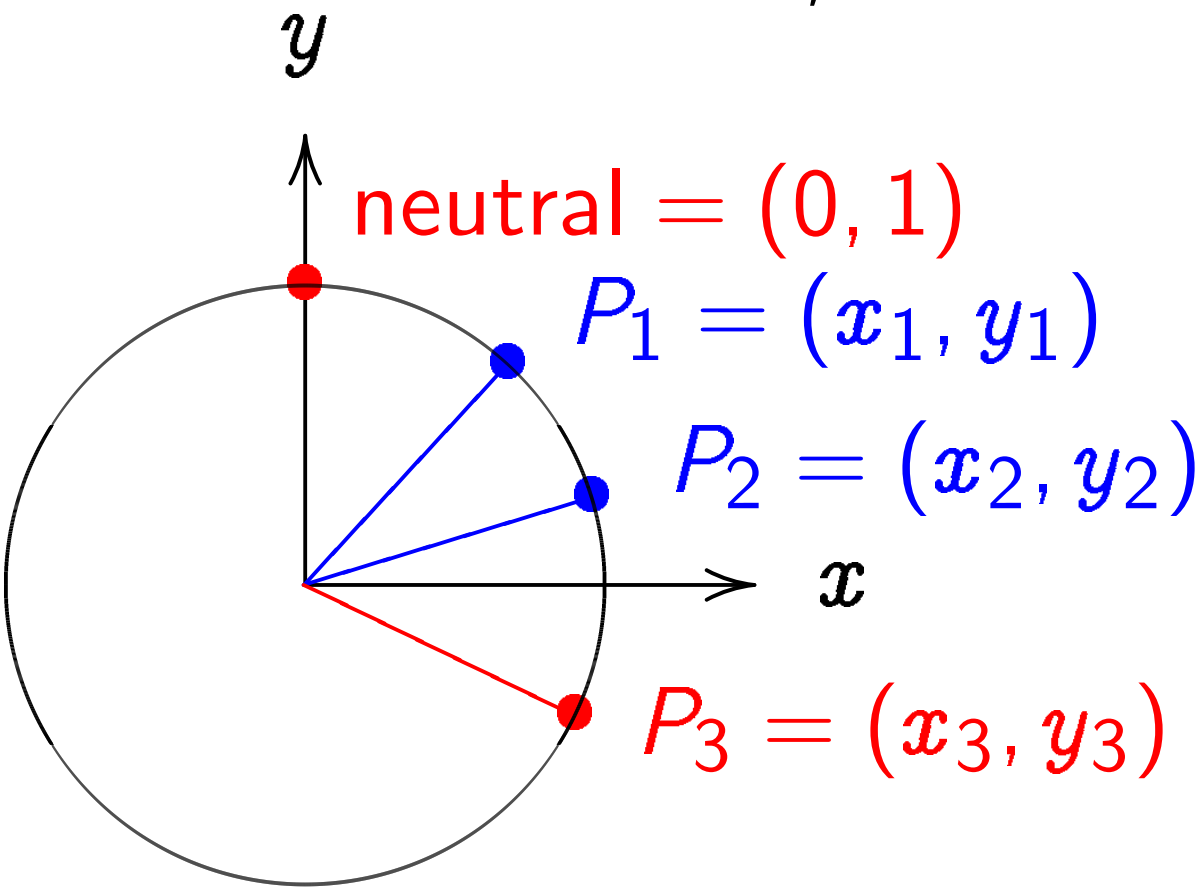
$x^2 + y^2 = 1$, parametrized by $x = \sin\alpha, \quad y = \cos\alpha$. Recall

$$(\sin(\alpha_1 + \alpha_2), \cos(\alpha_1 + \alpha_2)) =$$
$$(\sin\alpha_1 \cos\alpha_2 + \cos\alpha_1 \sin\alpha_2,$$
$$\cos\alpha_1 \cos\alpha_2 - \sin\alpha_1 \sin\alpha_2).$$

Adding two points corresponds
to adding the angles α_1 and α_2.
Angles modulo 360° are a group,
so points on clock are a group.

Neutral element: angle $\alpha = 0$;
point $(0, 1)$; "12:00".
The point with $\alpha = 180^\circ$
has order 2 and equals 6:00.
3:00 and 9:00 have order 4.
Inverse of point with α
is point with $-\alpha$
since $\alpha + (-\alpha) = 0$.
There are many more points
where angle α is not "nice."

Clock addition without sin, cos:



Use Cartesian coordinates for addition. Addition formula for the clock $x^2 + y^2 = 1$: sum of (x_1, y_1) and (x_2, y_2) is $(x_1y_2 + y_1x_2, y_1y_2 - x_1x_2)$.

Examples of clock addition:

$$\text{“2:00”} + \text{“5:00”}$$
$$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$$
$$= (-1/2, -\sqrt{3/4}) = \text{“7:00”}.$$

$$\text{“5:00”} + \text{“9:00”}$$
$$= (1/2, -\sqrt{3/4}) + (-1, 0)$$
$$= (\sqrt{3/4}, 1/2) = \text{“2:00”}.$$

$$2\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{24}{25}, \frac{7}{25}\right).$$

Examples of clock addition:

$$\text{“2:00”} + \text{“5:00”}$$
$$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$$
$$= (-1/2, -\sqrt{3/4}) = \text{“7:00”}.$$

$$\text{“5:00”} + \text{“9:00”}$$
$$= (1/2, -\sqrt{3/4}) + (-1, 0)$$
$$= (\sqrt{3/4}, 1/2) = \text{“2:00”}.$$

$$2\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{24}{25}, \frac{7}{25}\right).$$

$$3\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{117}{125}, \frac{-44}{125}\right).$$

Examples of clock addition:

$$\text{“2:00”} + \text{“5:00”}$$
$$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$$
$$= (-1/2, -\sqrt{3/4}) = \text{“7:00”}.$$

$$\text{“5:00”} + \text{“9:00”}$$
$$= (1/2, -\sqrt{3/4}) + (-1, 0)$$
$$= (\sqrt{3/4}, 1/2) = \text{“2:00”}.$$

$$2\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{24}{25}, \frac{7}{25}\right).$$

$$3\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{117}{125}, \frac{-44}{125}\right).$$

$$4\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{336}{625}, \frac{-527}{625}\right).$$

Examples of clock addition:

“2:00” + “5:00”

$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$

$= (-1/2, -\sqrt{3/4}) =$ “7:00”.

“5:00” + “9:00”

$= (1/2, -\sqrt{3/4}) + (-1, 0)$

$= (\sqrt{3/4}, 1/2) =$ “2:00”.

$$2\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{24}{25}, \frac{7}{25}\right).$$

$$3\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{117}{125}, \frac{-44}{125}\right).$$

$$4\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{336}{625}, \frac{-527}{625}\right).$$

$(x_1, y_1) + (0, 1) =$

Examples of clock addition:

“2:00” + “5:00”

$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$

$= (-1/2, -\sqrt{3/4}) =$ “7:00”.

“5:00” + “9:00”

$= (1/2, -\sqrt{3/4}) + (-1, 0)$

$= (\sqrt{3/4}, 1/2) =$ “2:00”.

$$2\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{24}{25}, \frac{7}{25}\right).$$

$$3\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{117}{125}, \frac{-44}{125}\right).$$

$$4\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{336}{625}, \frac{-527}{625}\right).$$

$$(x_1, y_1) + (0, 1) = (x_1, y_1).$$

Examples of clock addition:

“2:00” + “5:00”

$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$

$= (-1/2, -\sqrt{3/4}) =$ “7:00”.

“5:00” + “9:00”

$= (1/2, -\sqrt{3/4}) + (-1, 0)$

$= (\sqrt{3/4}, 1/2) =$ “2:00”.

$$2\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{24}{25}, \frac{7}{25}\right).$$

$$3\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{117}{125}, \frac{-44}{125}\right).$$

$$4\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{336}{625}, \frac{-527}{625}\right).$$

$(x_1, y_1) + (0, 1) = (x_1, y_1).$

$(x_1, y_1) + (-x_1, y_1) =$

Examples of clock addition:

“2:00” + “5:00”

$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$

$= (-1/2, -\sqrt{3/4}) =$ “7:00”.

“5:00” + “9:00”

$= (1/2, -\sqrt{3/4}) + (-1, 0)$

$= (\sqrt{3/4}, 1/2) =$ “2:00”.

$$2\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{24}{25}, \frac{7}{25}\right).$$

$$3\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{117}{125}, \frac{-44}{125}\right).$$

$$4\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{336}{625}, \frac{-527}{625}\right).$$

$$(x_1, y_1) + (0, 1) = (x_1, y_1).$$

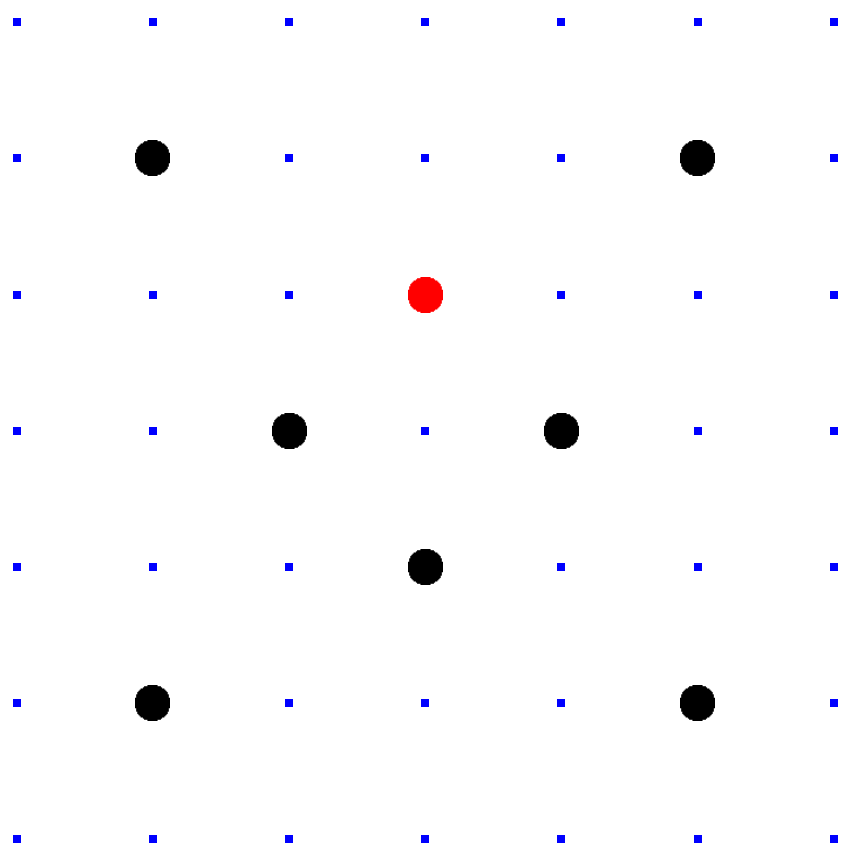
$$(x_1, y_1) + (-x_1, y_1) = (0, 1).$$

Problems

The coordinates
show a clear growth;
e.g. $625 = 5^4$
clearly shows the scalar 4.

Solution: Use modular reduction
as in Diffie-Hellman example.

Clocks over finite fields



$\mathrm{Clock}(\mathbf{F}_7) =$

$\{(x, y) \in \mathbf{F}_7 \times \mathbf{F}_7 : x^2 + y^2 = 1\}.$

Here $\mathbf{F}_7 = \{0, 1, 2, 3, 4, 5, 6\}$

$= \{0, 1, 2, 3, -3, -2, -1\}$

with $+, -, \times$ modulo 7.

Larger example: Clock($\mathbf{F}_{1000003}$).

Examples of clock addition:

$2(1000, 2) = (4000, 7)$.

$4(1000, 2) = (56000, 97)$.

$8(1000, 2) = (863970, 18817)$.

$16(1000, 2) = (549438, 156853)$.

$17(1000, 2) = (951405, 877356)$.

With 30 clock additions
we computed
$n(1000, 2) = (947472, 736284)$
for some 6-digit n.
Can you figure out n?

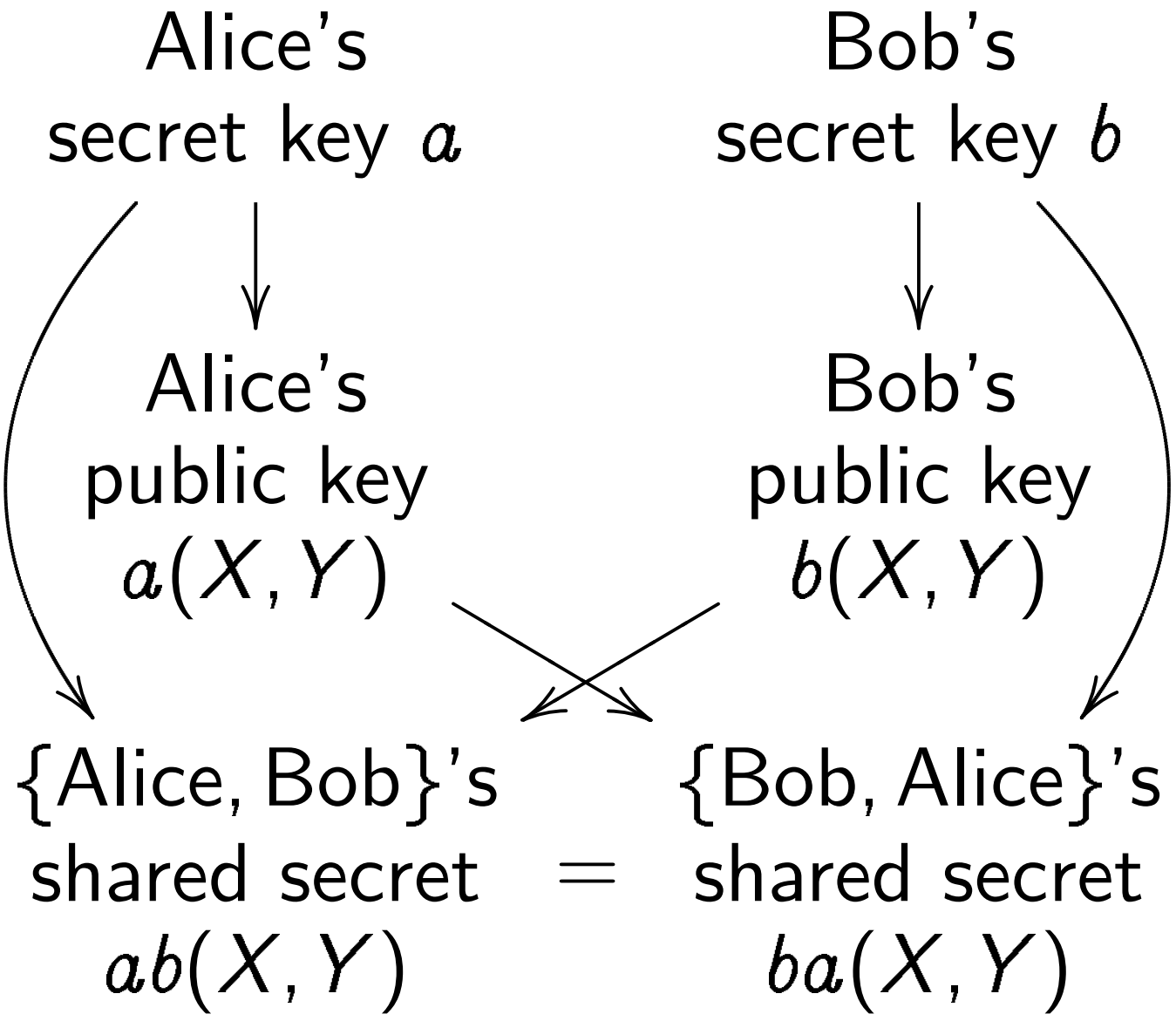
Clock cryptography

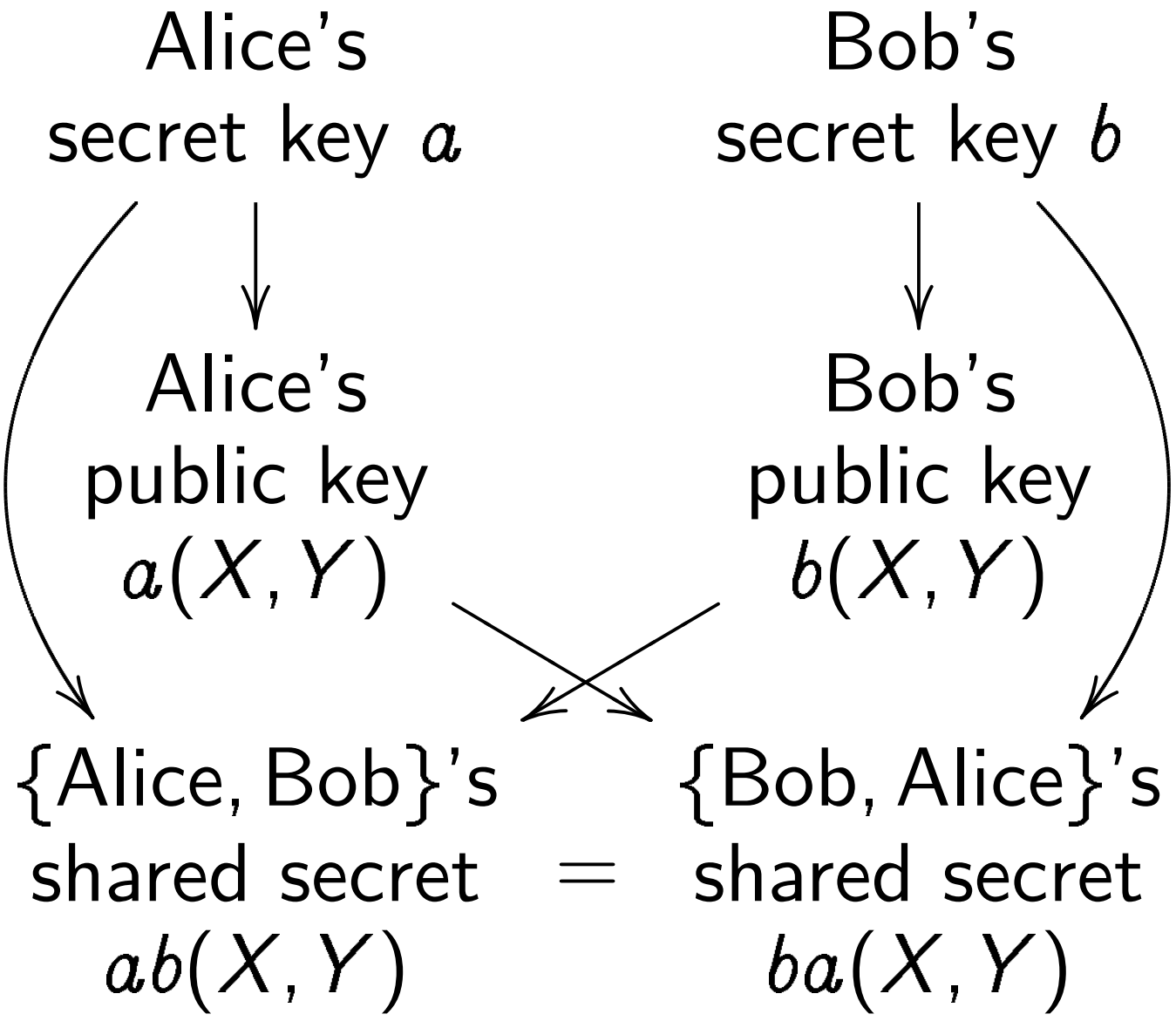
Standardize a large prime p
and some $(X, Y) \in \mathrm{Clock}(\mathbf{F}_p)$.
Follow standard security criteria.

Alice chooses big secret a.
Computes her public key $a(X, Y)$.

Bob chooses big secret b.
Computes his public key $b(X, Y)$.

Alice computes $a(b(X, Y))$.
Bob computes $b(a(X, Y))$.
They use this shared secret
to encrypt with AES-GCM etc.


Alice's
secret key a
Bob's
secret key b
Alice's
public key
$a(X, Y)$
Bob's
public key
$b(X, Y)$
{Alice, Bob}'s
shared secret
$ab(X, Y)$
=
{Bob, Alice}'s
shared secret
$ba(X, Y)$



Warning: Clocks aren't elliptic!

Can attack clock cryptography by combining congruences.

To match RSA-3072 security need $p \approx 2^{1536}$.

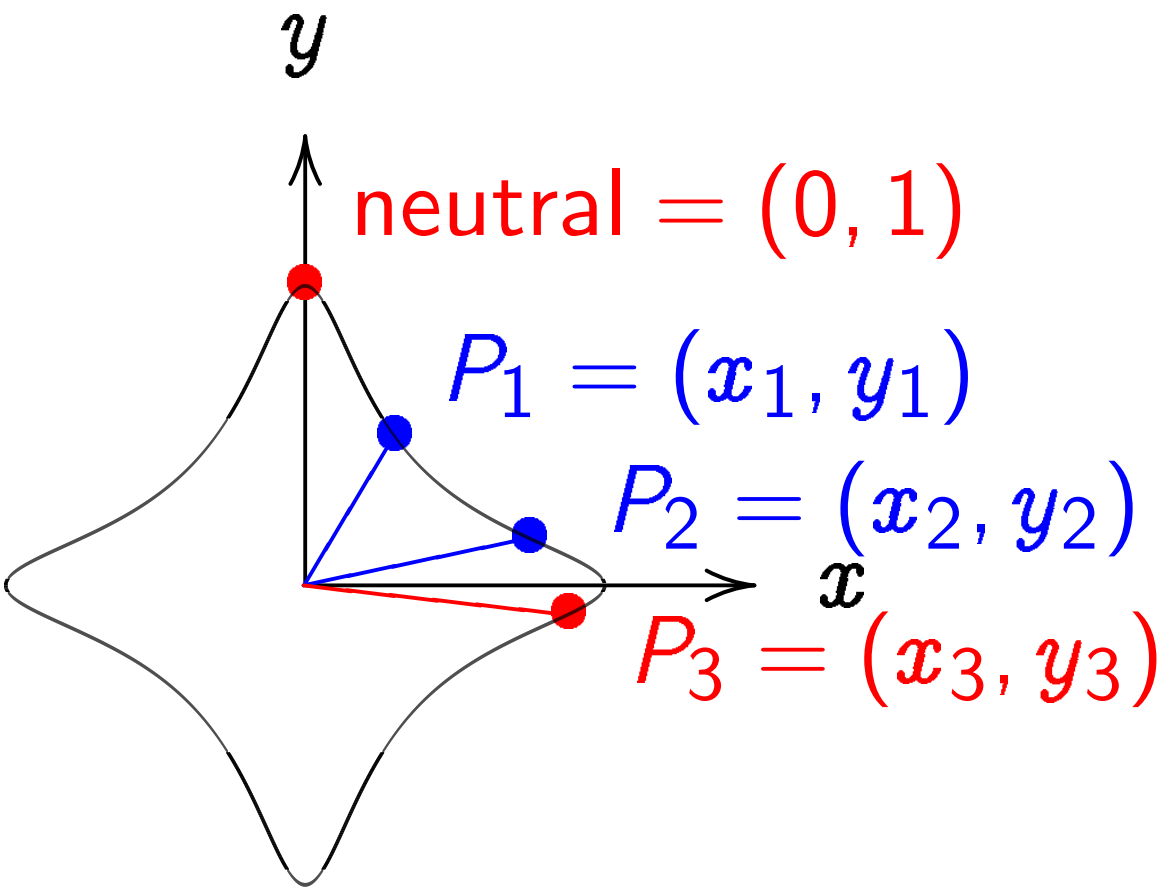
Exercise

How many multiplications
do you need to compute
$(x_1y_2 + y_1x_2, y_1y_2 - x_1x_2)$?

How many multiplications
do you need to double a point,
i.e. to compute
$(x_1y_1 + y_1x_1, y_1y_1 - x_1x_1)$?

How can you optimize the
computation if squarings are
cheaper than multiplications?
Assume $\mathbf{S} < \mathbf{M} < 2\mathbf{S}$.

Addition on an elliptic curve



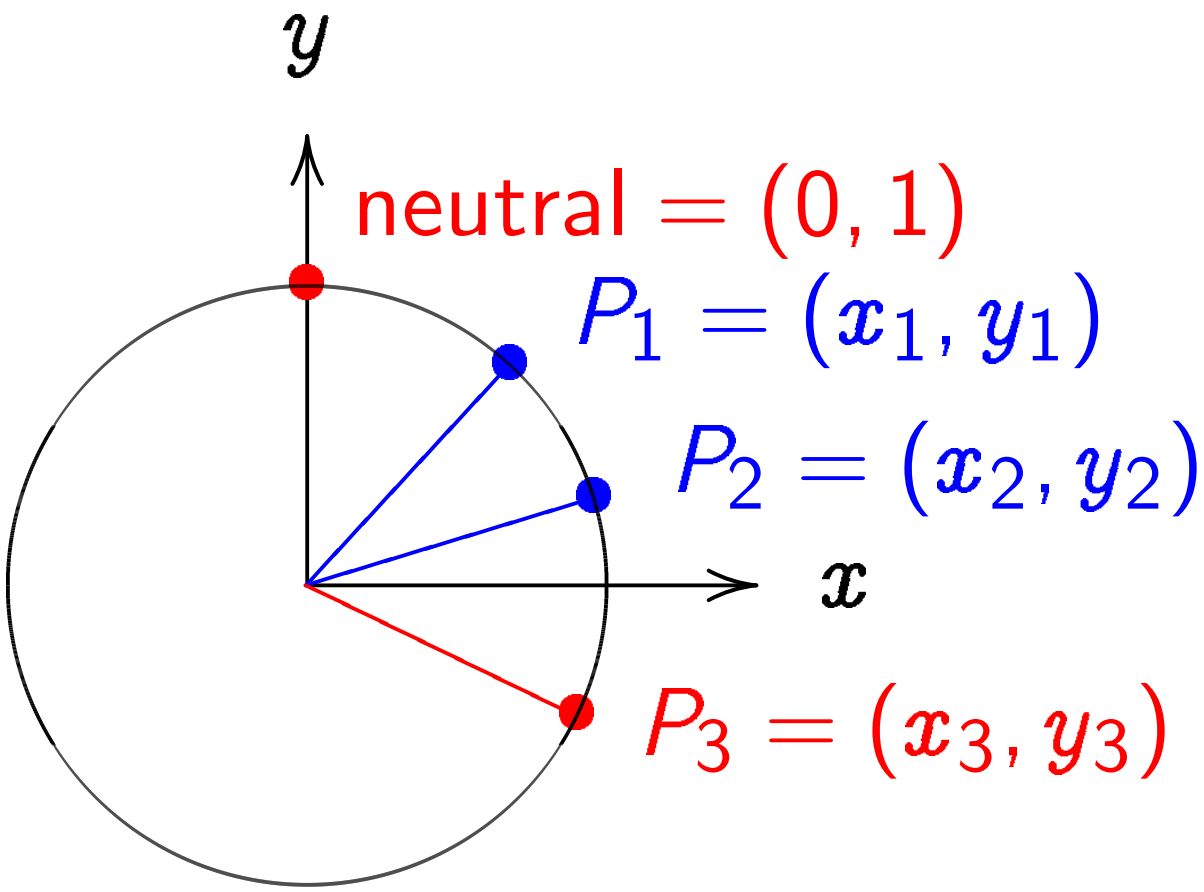
$x^2 + y^2 = 1 - 30x^2y^2$.

Sum of (x_1, y_1) and (x_2, y_2) is

$((x_1y_2+y_1x_2)/(1-30x_1x_2y_1y_2)$,

$(y_1y_2-x_1x_2)/(1+30x_1x_2y_1y_2))$.

The clock again, for comparison:



$x^2 + y^2 = 1$.

Sum of (x_1, y_1) and (x_2, y_2) is

$(x_1 y_2 + y_1 x_2,$

$y_1 y_2 - x_1 x_2)$.

“Hey, there were divisions
in the Edwards addition law!
What if the denominators are 0?”
Answer: They aren't!
If $x_i = 0$ or $y_i = 0$ then
$1 \pm 30x_1x_2y_1y_2 = 1 \neq 0$.
If $x^2 + y^2 = 1 - 30x^2y^2$
then $30x^2y^2 < 1$
so $\sqrt{30}\ |xy| < 1$.

"Hey, there were divisions
in the Edwards addition law!
What if the denominators are 0?"
Answer: They aren't!
If $x_i = 0$ or $y_i = 0$ then
$1 \pm 30x_1x_2y_1y_2 = 1 \neq 0$.
If $x^2 + y^2 = 1 - 30x^2y^2$
then $30x^2y^2 < 1$
so $\sqrt{30}\ |xy| < 1$.

If $x_1^2 + y_1^2 = 1 - 30x_1^2y_1^2$
and $x_2^2 + y_2^2 = 1 - 30x_2^2y_2^2$
then $\sqrt{30}\ |x_1y_1| < 1$
and $\sqrt{30}\ |x_2y_2| < 1$

"Hey, there were divisions
in the Edwards addition law!
What if the denominators are 0?"
Answer: They aren't!
If $x_i = 0$ or $y_i = 0$ then
$1 \pm 30x_1x_2y_1y_2 = 1 \neq 0$.
If $x^2 + y^2 = 1 - 30x^2y^2$
then $30x^2y^2 < 1$
so $\sqrt{30}\,|xy| < 1$.

If $x_1^2 + y_1^2 = 1 - 30x_1^2y_1^2$
and $x_2^2 + y_2^2 = 1 - 30x_2^2y_2^2$
then $\sqrt{30}\,|x_1y_1| < 1$
and $\sqrt{30}\,|x_2y_2| < 1$
so $30\,|x_1y_1x_2y_2| < 1$
so $1 \pm 30x_1x_2y_1y_2 > 0$.

The Edwards addition law
$(x_1, y_1) + (x_2, y_2) =$
$((x_1y_2+y_1x_2)/(1-30x_1x_2y_1y_2),$
$(y_1y_2-x_1x_2)/(1+30x_1x_2y_1y_2))$
is a group law for the curve
$x^2 + y^2 = 1 - 30x^2y^2$.

Some calculation required:

addition result is on curve;

addition law is associative.

Other parts of proof are easy:

addition law is commutative;

$(0, 1)$ is neutral element;

$(x_1, y_1) + (-x_1, y_1) = (0, 1)$.

More Edwards curves

Fix an odd prime power q.

Fix a *non-square* $d \in \mathbf{F}_q$.

$$\{(x, y) \in \mathbf{F}_q \times \mathbf{F}_q : \\ x^2 + y^2 = 1 + dx^2y^2\}$$

is a commutative group with

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

defined by Edwards addition law:

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

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

Denominators are never 0.

But need different proof;

"$x^2 + y^2 > 0$" doesn't work.

Denominators are never 0.

But need different proof;
"$x^2 + y^2 > 0$" doesn't work.

If $x_1^2 + y_1^2 = 1 + dx_1^2 y_1^2$
and $x_2^2 + y_2^2 = 1 + dx_2^2 y_2^2$
and $dx_1 x_2 y_1 y_2 = \pm 1$

Denominators are never 0.

But need different proof;

"$x^2 + y^2 > 0$" doesn't work.

If $x_1^2 + y_1^2 = 1 + dx_1^2 y_1^2$

and $x_2^2 + y_2^2 = 1 + dx_2^2 y_2^2$

and $dx_1 x_2 y_1 y_2 = \pm 1$

then $dx_1^2 y_1^2 (x_2 + y_2)^2$

$= dx_1^2 y_1^2 (x_2^2 + y_2^2 + 2x_2 y_2)$

Denominators are never 0.

But need different proof;

"$x^2 + y^2 > 0$" doesn't work.

If $x_1^2 + y_1^2 = 1 + dx_1^2 y_1^2$

and $x_2^2 + y_2^2 = 1 + dx_2^2 y_2^2$

and $dx_1 x_2 y_1 y_2 = \pm 1$

then $dx_1^2 y_1^2 (x_2 + y_2)^2$

$= dx_1^2 y_1^2 (x_2^2 + y_2^2 + 2x_2 y_2)$

$= dx_1^2 y_1^2 (dx_2^2 y_2^2 + 1 + 2x_2 y_2)$

Denominators are never 0.

But need different proof;

"$x^2 + y^2 > 0$" doesn't work.

If $x_1^2 + y_1^2 = 1 + dx_1^2 y_1^2$

and $x_2^2 + y_2^2 = 1 + dx_2^2 y_2^2$

and $dx_1 x_2 y_1 y_2 = \pm 1$

then $dx_1^2 y_1^2 (x_2 + y_2)^2$

$= dx_1^2 y_1^2 (x_2^2 + y_2^2 + 2x_2 y_2)$

$= dx_1^2 y_1^2 (dx_2^2 y_2^2 + 1 + 2x_2 y_2)$

$= d^2 x_1^2 y_1^2 x_2^2 y_2^2 + dx_1^2 y_1^2 + 2dx_1^2 y_1^2 x_2 y_2$

Denominators are never 0.

But need different proof;

"$x^2 + y^2 > 0$" doesn't work.

If $x_1^2 + y_1^2 = 1 + dx_1^2 y_1^2$

and $x_2^2 + y_2^2 = 1 + dx_2^2 y_2^2$

and $dx_1 x_2 y_1 y_2 = \pm 1$

then $dx_1^2 y_1^2 (x_2 + y_2)^2$

$= dx_1^2 y_1^2 (x_2^2 + y_2^2 + 2x_2 y_2)$

$= dx_1^2 y_1^2 (dx_2^2 y_2^2 + 1 + 2x_2 y_2)$

$= d^2 x_1^2 y_1^2 x_2^2 y_2^2 + dx_1^2 y_1^2 + 2dx_1^2 y_1^2 x_2 y_2$

$= 1 + dx_1^2 y_1^2 \pm 2x_1 y_1$

Denominators are never 0.

But need different proof;

"$x^2 + y^2 > 0$" doesn't work.

If $x_1^2 + y_1^2 = 1 + dx_1^2 y_1^2$

and $x_2^2 + y_2^2 = 1 + dx_2^2 y_2^2$

and $dx_1 x_2 y_1 y_2 = \pm 1$

then $dx_1^2 y_1^2 (x_2 + y_2)^2$

$= dx_1^2 y_1^2 (x_2^2 + y_2^2 + 2x_2 y_2)$

$= dx_1^2 y_1^2 (dx_2^2 y_2^2 + 1 + 2x_2 y_2)$

$= d^2 x_1^2 y_1^2 x_2^2 y_2^2 + dx_1^2 y_1^2 + 2dx_1^2 y_1^2 x_2 y_2$

$= 1 + dx_1^2 y_1^2 \pm 2x_1 y_1$

$= x_1^2 + y_1^2 \pm 2x_1 y_1$

Denominators are never 0.

But need different proof;

"$x^2 + y^2 > 0$" doesn't work.

If $x_1^2 + y_1^2 = 1 + dx_1^2 y_1^2$

and $x_2^2 + y_2^2 = 1 + dx_2^2 y_2^2$

and $dx_1 x_2 y_1 y_2 = \pm 1$

then $dx_1^2 y_1^2 (x_2 + y_2)^2$

$= dx_1^2 y_1^2 (x_2^2 + y_2^2 + 2x_2 y_2)$

$= dx_1^2 y_1^2 (dx_2^2 y_2^2 + 1 + 2x_2 y_2)$

$= d^2 x_1^2 y_1^2 x_2^2 y_2^2 + dx_1^2 y_1^2 + 2dx_1^2 y_1^2 x_2 y_2$

$= 1 + dx_1^2 y_1^2 \pm 2x_1 y_1$

$= x_1^2 + y_1^2 \pm 2x_1 y_1$

$= (x_1 \pm y_1)^2.$

Case 1: $x_2 + y_2 \neq 0$. Then

$$d = \left(\frac{x_1 \pm y_1}{x_1 y_1 (x_2 + y_2)} \right)^2,$$

contradiction.

Case 1: $x_2 + y_2 \neq 0$. Then

$$d = \left(\frac{x_1 \pm y_1}{x_1 y_1 (x_2 + y_2)} \right)^2,$$

contradiction.

Case 2: $x_2 - y_2 \neq 0$. Then

$$d = \left(\frac{x_1 \mp y_1}{x_1 y_1 (x_2 - y_2)} \right)^2,$$

contradiction.

Case 1: $x_2 + y_2 \neq 0$. Then

$$d = \left(\frac{x_1 \pm y_1}{x_1 y_1 (x_2 + y_2)} \right)^2,$$

contradiction.

Case 2: $x_2 - y_2 \neq 0$. Then

$$d = \left(\frac{x_1 \mp y_1}{x_1 y_1 (x_2 - y_2)} \right)^2,$$

contradiction.

Case 3: $x_2 + y_2 = x_2 - y_2 = 0$. Then $x_2 = 0$ and $y_2 = 0$, contradiction.

Group operations

Can compute on Edwards curve,
do Diffie–Hellman key exchange.
Formulas use divisions.
Denominators are nonzero but
divisions are expensive.
Better: postpone divisions
and work with fractions.
$A = Z_1 \cdot Z_2$; $B = A^2$; $C = X_1 \cdot X_2$; $D = Y_1 \cdot Y_2$; $E = d \cdot C \cdot D$; $F = B - E$; $G = B + E$; $X_3 = A \cdot F \cdot ((X_1 + Y_1) \cdot (X_2 + Y_2) - C - D)$; $Y_3 = A \cdot G \cdot (D - C)$; $Z_3 = F \cdot G$.
Needs $1\mathbf{S}+10\mathbf{M}+1\mathbf{M}_d$.

Edwards curves are elliptic!

Can use Edwards group in crypto.
. . . if it's a "strong" curve.
Need to compute group order.
If no large prime factor in order,
must switch to another d;
this very often happens.

Also check "twist security,"
"embedding degree," et al.

IEEE Std 1363 is good source
for most security criteria
except twist security.

Safe example, "Curve25519":
$q = 2^{255} - 19$; $d = 1 - 1/121666$.

Using ECC sensibly

Typical starting point:
Client knows secret key a
and server's public key $b(X, Y)$.
Client computes (and caches)
shared secret $ab(X, Y)$.

Client has packet for server.
Generates unique nonce.
Uses shared secret to encrypt
and authenticate packet.

Total packet overhead:
24 bytes for nonce,
16 bytes for authenticator,
32 bytes for client's public key.

Server receives packet,
sees client's public key $a(X, Y)$.
Server computes (and caches)
shared secret $ab(X, Y)$.

Server uses shared secret
to verify authenticator
and decrypt packet.

Client and server encrypt,
authenticate, verify, and decrypt
all subsequent packets
in the same way,
using the same shared secret.

Easy-to-use packet protection:
`crypto_box` from
`nacl.cace-project.eu`.

High-security curve (Curve25519).
High-security implementation
(e.g., no secret array indices).
Extensive code validation.

Server can compute shared secrets
for 1000000 new clients
in 40 seconds of computation
on a Core 2 Quad.

Now you are ready to run software
using elliptic curves. But there is
more to know.

More curves

Can we use Edwards curve
$x^2 + y^2 = 1 + dx^2y^2$
when d is a square?

$d = 0$: Clock. Not very secure.

$d = 1$: Even worse problems.

Other squares d:
The Edwards curve *is* elliptic
but it is not "complete."
Need "points at ∞." These
are the points where x or y has
division by 0.

Example of how ∞ appears:

Define $d = 4/49 = (2/7)^2$.

$(4, 7)$ is a point
on $x^2 + y^2 = 1 + dx^2y^2$.

$(-7/8, 1/2)$ is a point
on $x^2 + y^2 = 1 + dx^2y^2$.

Try adding these points:

$$x_3 = \frac{4 \cdot \frac{1}{2} - 7 \cdot \frac{7}{8}}{1 - \frac{4}{49} \cdot 4 \cdot \frac{7}{8} \cdot 7 \cdot \frac{1}{2}} = \frac{\frac{-33}{8}}{0},$$

$$y_3 = \frac{7 \cdot \frac{1}{2} + 4 \cdot \frac{7}{8}}{1 + \frac{4}{49} \cdot 4 \cdot \frac{7}{8} \cdot 7 \cdot \frac{1}{2}} = \frac{7}{2}.$$

New definition of set of curve points when d is a square:

$$\{(x,y) : x^2 + y^2 = 1 + dx^2y^2\}$$
$$\cup \left\{(\pm 1/\sqrt{d}, \infty)\right\}$$
$$\cup \left\{(\infty, \pm 1/\sqrt{d})\right\}.$$

Even more trouble:

Again take $d = 4/49 = (2/7)^2$.

$(4, 7)$ is a point
on $x^2 + y^2 = 1 + dx^2y^2$.

$(7/8, 1/2)$ is a point
on $x^2 + y^2 = 1 + dx^2y^2$.

Try adding these points:

$$x_3 = \frac{4 \cdot \frac{1}{2} + 7 \cdot \frac{7}{8}}{1 + \frac{4}{49} \cdot 4 \cdot \frac{7}{8} \cdot 7 \cdot \frac{1}{2}} = \frac{65}{16},$$

$$y_3 = \frac{7 \cdot \frac{1}{2} - 4 \cdot \frac{7}{8}}{1 - \frac{4}{49} \cdot 4 \cdot \frac{7}{8} \cdot 7 \cdot \frac{1}{2}} = \frac{0}{0}.$$

Generalize addition law:

Represent (x_i, y_i) by
$(X_i/Z_i, Y_i/T_i)$ and use

$$(X_1/Z_1, Y_1/T_1) +$$
$$(X_2/Z_2, Y_2/T_2) =$$
$$\big((X_1Y_2Z_2T_1 + X_2Y_1Z_1T_2)/$$
$$(Z_1Z_2T_1T_2 + dX_1X_2Y_1Y_2),$$
$$(Y_1Y_2Z_1Z_2 - aX_1X_2T_1T_2)/$$
$$(Z_1Z_2T_1T_2 - dX_1X_2Y_1Y_2)\big)$$

if defined;

or

$(X_1Y_1Z_2T_2 + X_2Y_2Z_1T_1/$
$X_1X_2T_1T_2 + Y_1Y_2Z_1Z_2),$
$(X_1Y_1Z_2T_2 - X_2Y_2Z_1T_1)/$
$(X_1Y_2Z_2T_1 - X_2Y_1Z_1T_2))$
if defined.

Have shown in ePrint 2009/580 that at least one of these two expressions is defined for any pair of input points.
Have 2 addition laws to cover all inputs even in the incomplete case where d is a square.
As a designer can choose parameters and choose d not to be a square.

The second law is interesting also outside the context of square values of d. Hisil et al. at Asiacrypt 2008 obtained better addition speed by using $(x_1, y_1) + (x_2, y_2) = \left(\frac{x_1y_1+x_2y_2}{x_1x_2+y_1y_2}, \frac{x_1y_1-x_2y_2}{x_1y_2-x_2y_1}\right)$.

Attention: these formulas fail for doubling.

Curious fact: formulas do not involve curve parameter d

Twisted Edwards curves

Generalization to cover more curves over given finite field $\mathbf{F}_q$: Use $a, d \in \mathbf{F}_q^*$ with $a \neq d$ and consider twisted Edwards curve $ax^2 + y^2 = 1 + dx^2y^2$.

Particular fast choice: $a = -1$ gives additions in $8\mathbf{M}$.

There are many perspectives on elliptic-curve computations.
Early development:

1984 (published 1987) Lenstra:
ECM, the elliptic-curve method
of factoring integers.

1984 (published 1985) Miller,
and independently
1984 (published 1987) Koblitz:
Elliptic-curve cryptography.

Bosma, Goldwasser–Kilian,
Chudnovsky–Chudnovsky, Atkin:
elliptic-curve primality proving.

The Edwards perspective is new!

1761 Euler, 1866 Gauss
introduced an addition law
for $x^2 + y^2 = 1 - x^2y^2$,
the "lemniscatic elliptic curve."

2007 Edwards generalized to
many curves $x^2+y^2 = 1+c^4x^2y^2$.
Theorem: have now obtained
all elliptic curves over $\overline{\mathbf{Q}}$.

2007 Bernstein–Lange:
Edwards addition law is complete
for $x^2 + y^2 = 1 + dx^2y^2$ if $d \neq$■;
and gives new ECC speed records.

Representing curve points

Crypto 1985, Miller, "Use of elliptic curves in cryptography":

Given $n \in \mathbf{Z}$, $P \in E(\mathbf{F}_q)$, division-polynomial recurrence computes $nP \in E(\mathbf{F}_q)$
"in $26 \log_2 n$ multiplications";
but can do better!

"It appears to be best to represent the points on the curve in the following form:
Each point is represented by the triple (x, y, z) which corresponds to the point $(x/z^2, y/z^3)$."

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."

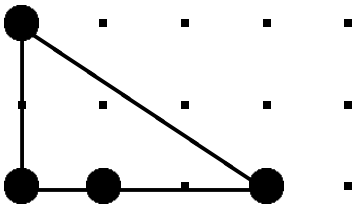
Short Weierstrass:
$y^2 = x^3 + ax + b$.

Jacobi intersection:
$s^2 + c^2 = 1$, $as^2 + d^2 = 1$.

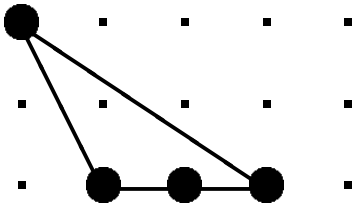
Jacobi quartic: $y^2 = x^4 + 2ax^2 + 1$.

Hessian: $x^3 + y^3 + 1 = 3dxy$.

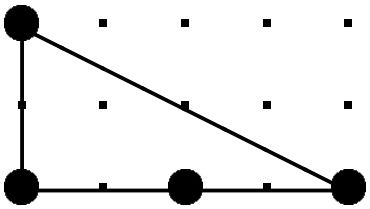
Some Newton polygons



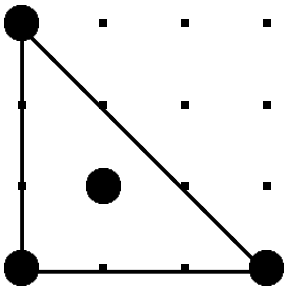
Short Weierstrass



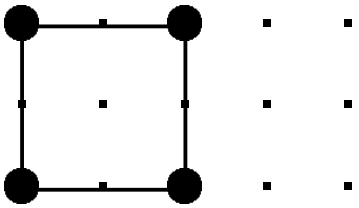
Montgomery



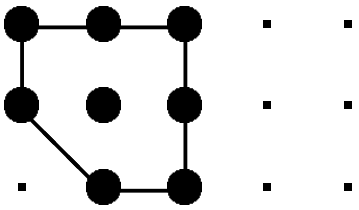
Jacobi quartic



Hessian



Edwards



Binary Edwards

Birational equivalence

Starting from point (x, y)
on $x^2 + y^2 = 1 + dx^2y^2$:

Define $A = 2(1 + d)/(1 - d)$,
$B = 4/(1 - d)$;
$u = (1 + y)/(B(1 - y))$,
$v = u/x = (1 + y)/(Bx(1 - y))$.
(Skip a few exceptional points.)

Then (u, v) is a point on
a long Weierstrass curve:
$v^2 = u^3 + (A/B)u^2 + (1/B^2)u$;

Easily invert this map:
$x = u/v$, $y = (Bu - 1)/(Bu + 1)$.
$\Rightarrow$ Same discrete-log security!

Optimizing Jacobian coordinates

For "traditional" $(X/Z^2, Y/Z^3)$
on $y^2 = x^3 + ax + b$:
1986 Chudnovsky–Chudnovsky
state explicit formulas using
10**M** for DBL; 16**M** for ADD.

Consequence:
$$\approx \left(10 \lg n + 16 \frac{\lg n}{\lg \lg n}\right) \mathbf{M}$$
to compute $n, P \mapsto nP$
using sliding-windows method
of scalar multiplication.

Notation: $\lg = \log_2$.

Squaring is faster than **M**.

Here are the DBL formulas:

$$S = 4X_1 \cdot Y_1^2;$$
$$M = 3X_1^2 + aZ_1^4;$$
$$T = M^2 - 2S;$$
$$X_3 = T;$$
$$Y_3 = M \cdot (S - T) - 8Y_1^4;$$
$$Z_3 = 2Y_1 \cdot Z_1.$$

Total cost $3\mathbf{M} + 6\mathbf{S} + 1\mathbf{D}$ where
S is the cost of squaring in $\mathbf{F}_q$,
D is the cost of multiplying by a.

The squarings produce
$X_1^2, Y_1^2, Y_1^4, Z_1^2, Z_1^4, M^2$.

Most ECC standards choose
curves that make formulas faster.

Curve-choice advice from
1986 Chudnovsky–Chudnovsky:

Can eliminate the 1**D**
by choosing curve with $a = 1$.

But "it is even smarter"
to choose curve with $a = -3$.

If $a = -3$ then $M = 3(X_1^2 - Z_1^4)$
$= 3(X_1 - Z_1^2) \cdot (X_1 + Z_1^2)$.
Replace 2**S** with 1**M**.

Now DBL costs 4**M** + 4**S**.

2001 Bernstein:
$3\mathbf{M} + 5\mathbf{S}$ for DBL.
$11\mathbf{M} + 5\mathbf{S}$ for ADD.

How? Easy $\mathbf{S} - \mathbf{M}$ tradeoff:
instead of computing $2Y_1 \cdot Z_1$,
compute $(Y_1 + Z_1)^2 - Y_1^2 - Z_1^2$.
DBL formulas were already
computing Y_1^2 and Z_1^2.

Same idea for the ADD formulas,
but have to scale X, Y, Z
to eliminate divisions by 2.

ADD for $y^2 = x^3 + ax + b$:
$U_1 = X_1 Z_2^2$, $U_2 = X_2 Z_1^2$,
$S_1 = Y_1 Z_2^3$, $S_2 = Y_2 Z_1^3$,
many more computations.

1986 Chudnovsky–Chudnovsky:
"We suggest to write
addition formulas involving
(X, Y, Z, Z^2, Z^3)."

Disadvantages:
Allocate space for Z^2, Z^3.
Pay $1\mathbf{S}+1\mathbf{M}$ in ADD and in DBL.

Advantages:
Save $2\mathbf{S} + 2\mathbf{M}$ at start of ADD.
Save $1\mathbf{S}$ at start of DBL.

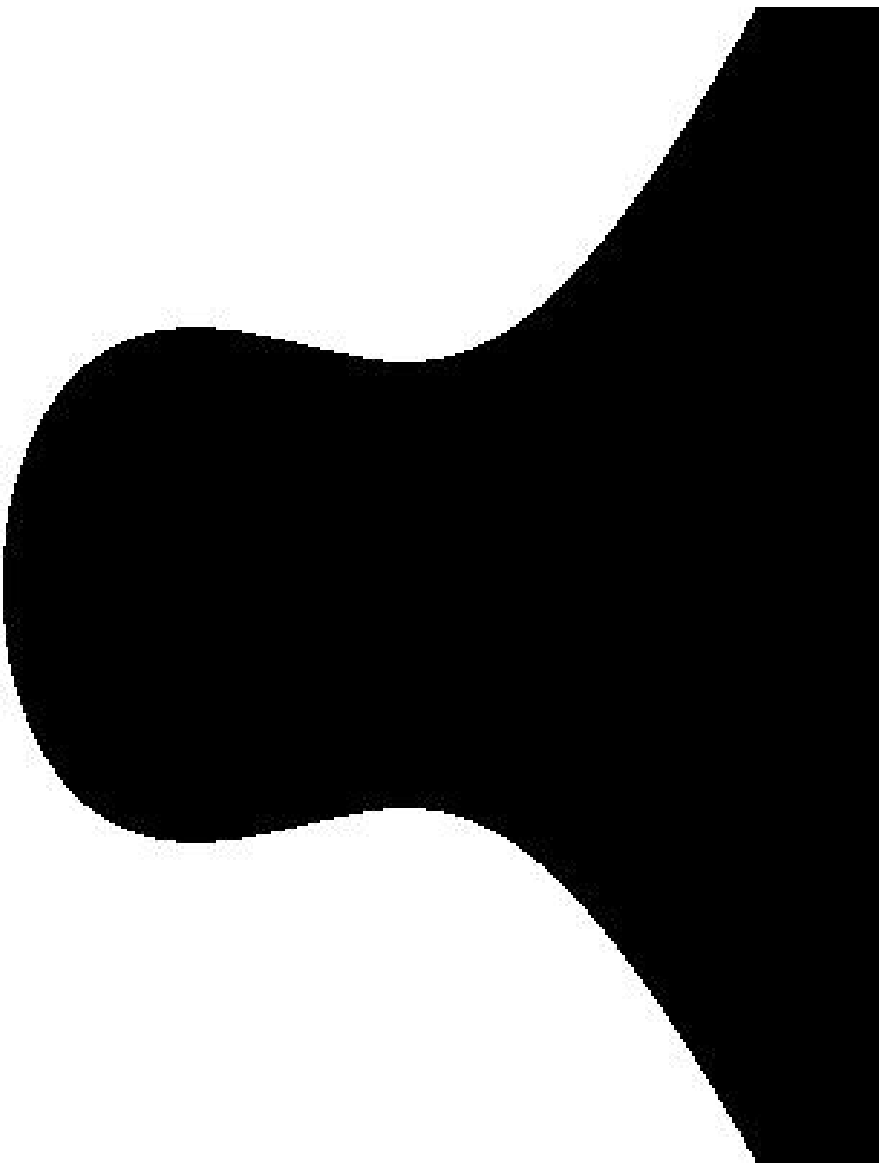
1998 Cohen–Miyaji–Ono:
Store point as $(X : Y : Z)$.
If point is input to ADD,
also cache Z^2 and Z^3.
No cost, aside from space.
If point is input to another ADD,
reuse Z^2, Z^3. Save $1\mathbf{S} + 1\mathbf{M}$!

Best Jacobian speeds today,
including $\mathbf{S} - \mathbf{M}$ tradeoffs:
$3\mathbf{M} + 5\mathbf{S}$ for DBL if $a = -3$.
$11\mathbf{M} + 5\mathbf{S}$ for ADD.
$10\mathbf{M} + 4\mathbf{S}$ for reADD.
$7\mathbf{M} + 4\mathbf{S}$ for mADD (i.e. $Z_2 = 1$).

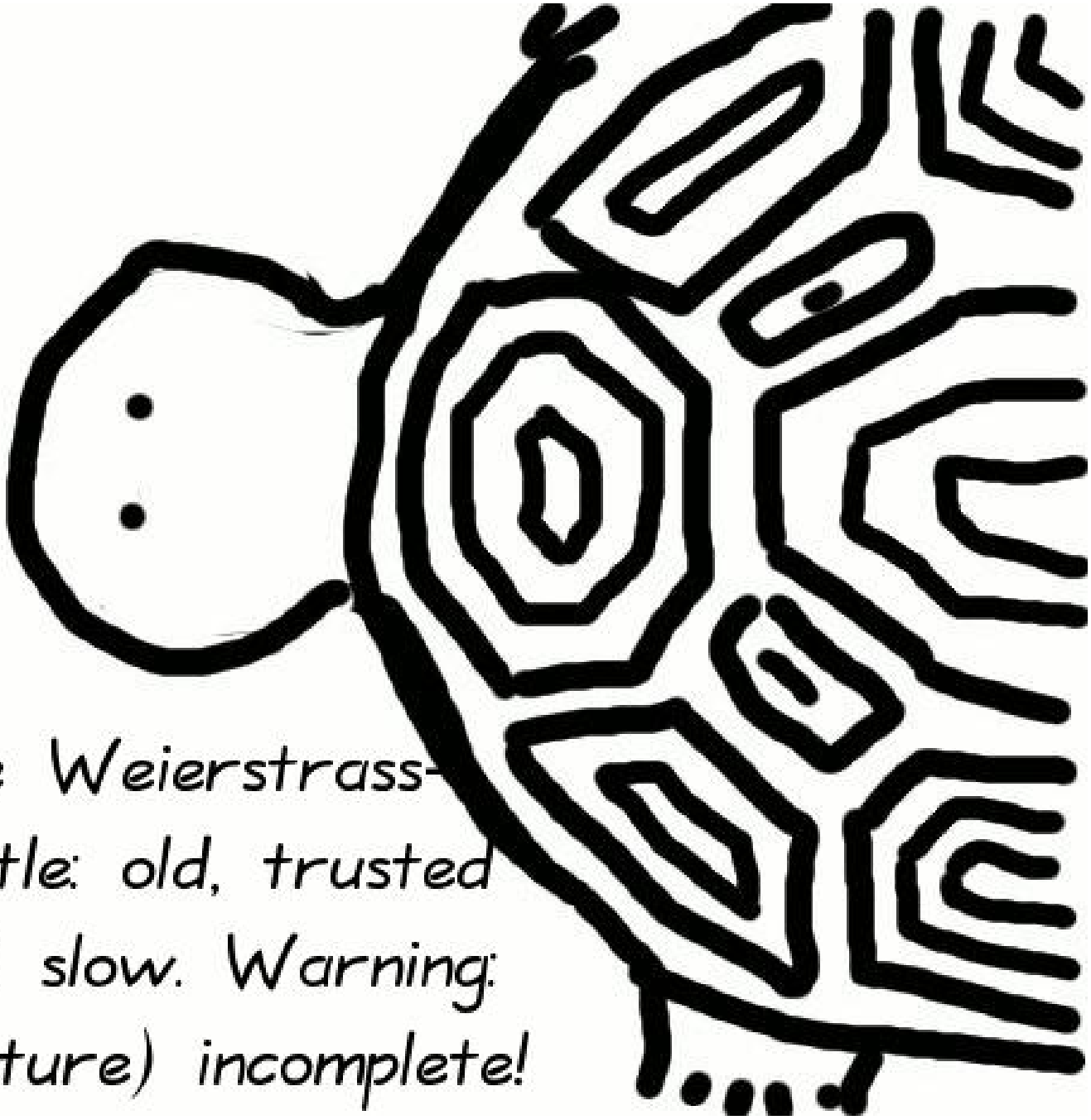
Compare to speeds for Edwards curves $x^2 + y^2 = 1 + dx^2y^2$ in projective coordinates (2007 Bernstein–Lange):
$3\mathbf{M} + 4\mathbf{S}$ for DBL.
$10\mathbf{M} + 1\mathbf{S} + 1\mathbf{D}$ for ADD.
$9\mathbf{M} + 1\mathbf{S} + 1\mathbf{D}$ for mADD.

Inverted Edwards coordinates (2007 Bernstein–Lange):
$3\mathbf{M} + 4\mathbf{S} + 1\mathbf{D}$ for DBL.
$9\mathbf{M} + 1\mathbf{S} + 1\mathbf{D}$ for ADD.
$8\mathbf{M} + 1\mathbf{S} + 1\mathbf{D}$ for mADD.

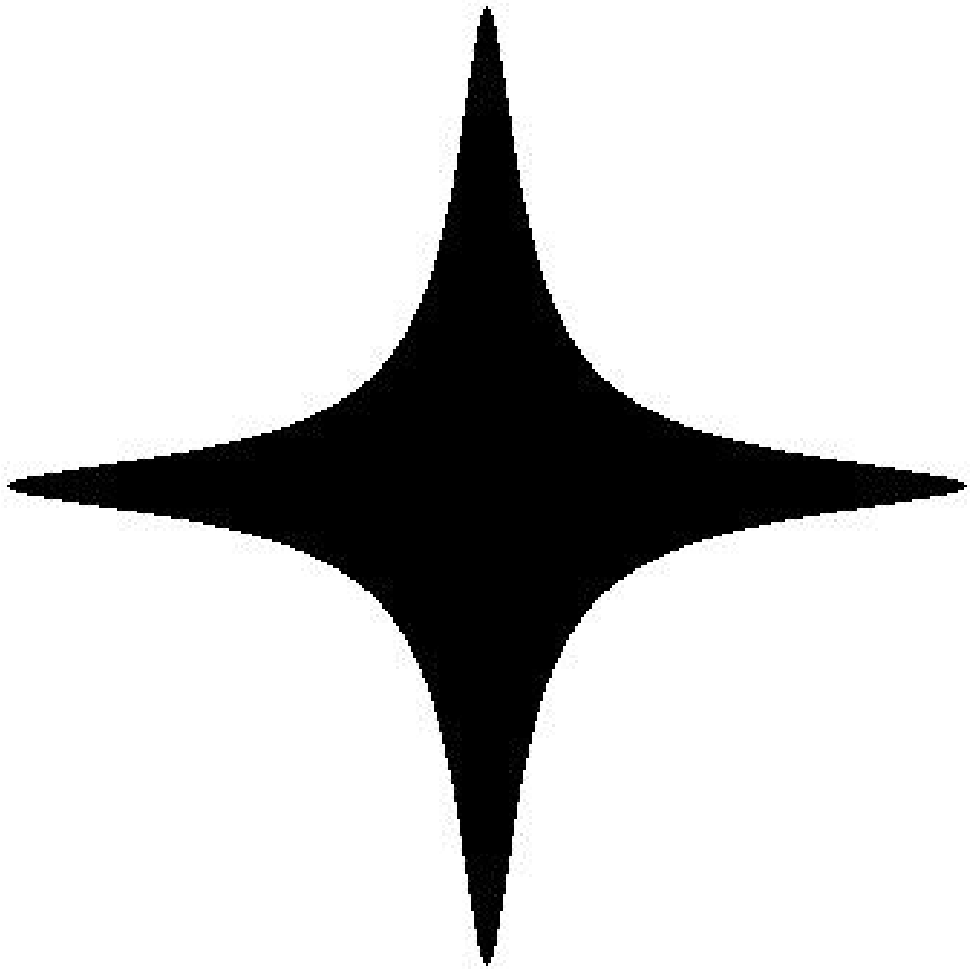
Even better speeds from extended/completed coordinates (2008 Hisil–Wong–Carter–Dawson).



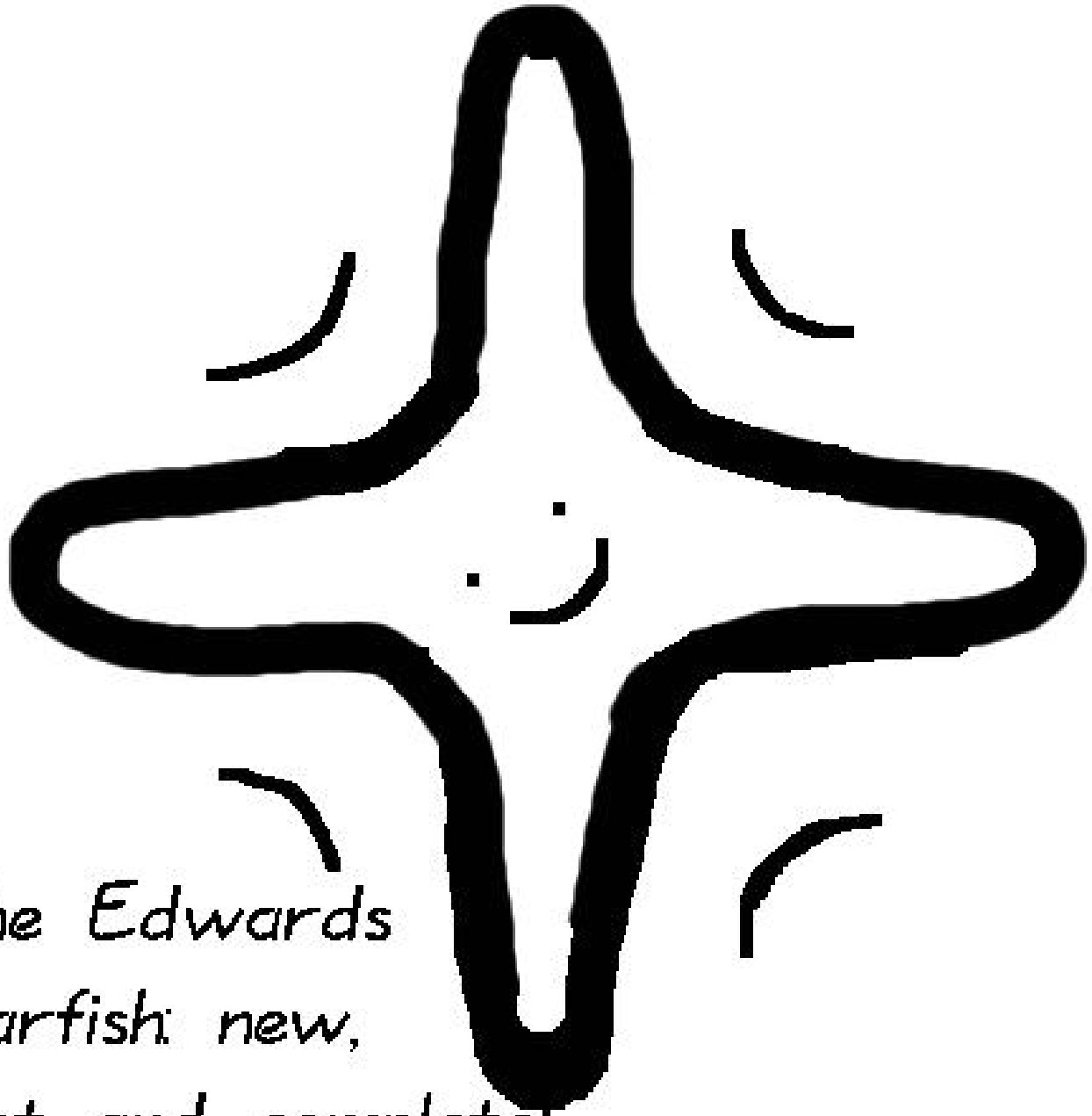
$$y^2 = x^3 - 0.4x + 0.7$$



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!

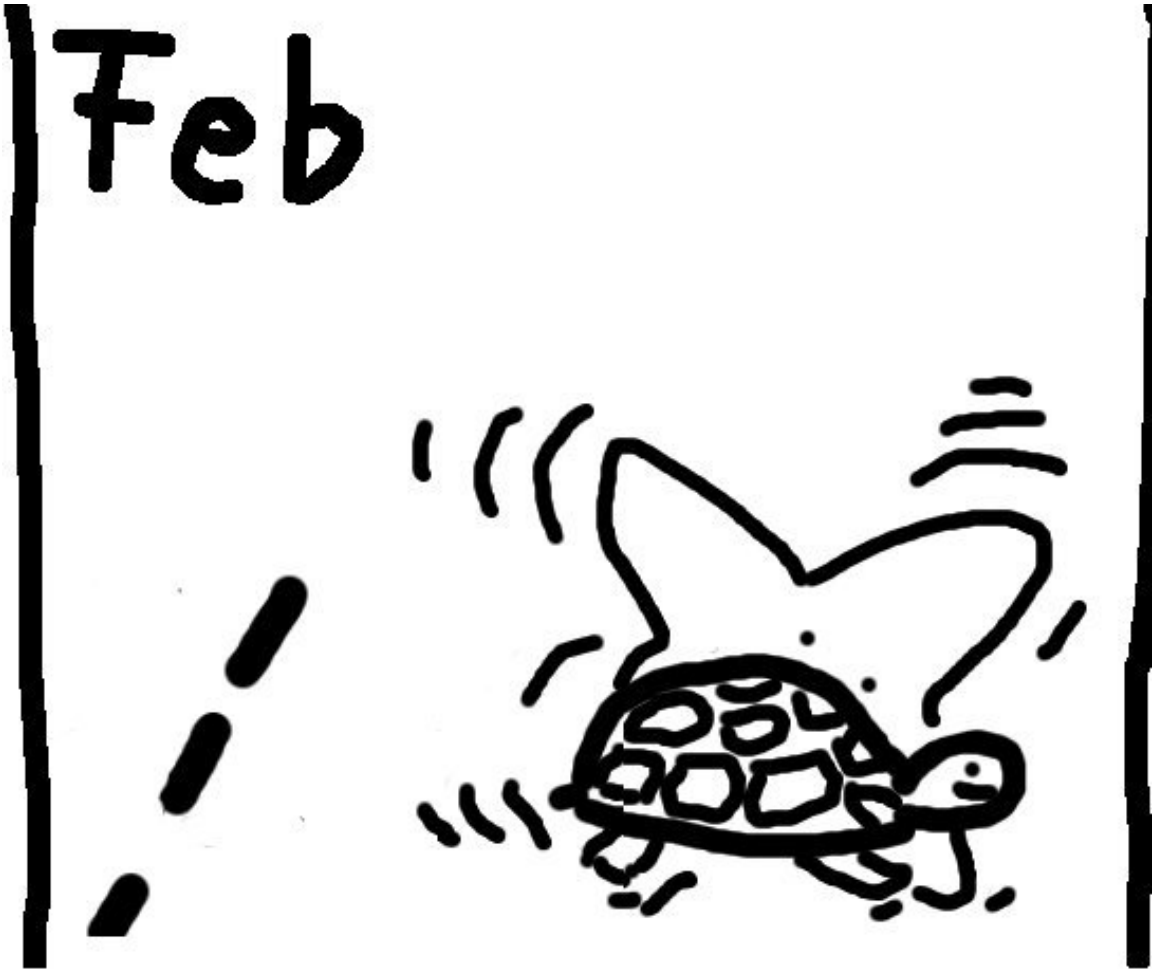
Start!

1985

Weierstrass sets off. Edwards
left behind sleeping

2007-Jan

Weierstrass has made some progress -
finally Edwards wakes up.



Exciting progress: Edwards about to overtake!!

Mar

And the winner is: Edwards!

Speed-oriented Jacobian standards

2000 IEEE "Std 1363"
uses Weierstrass curves
in Jacobian coordinates
to "provide the fastest
arithmetic on elliptic curves."
Also specifies a method of
choosing curves $y^2 = x^3 - 3x + b$.

2000 NIST "FIPS 186–2"
standardizes five such curves.

2005 NSA "Suite B" recommends
two of the NIST curves as
the only public-key cryptosystems
for U.S. government use.

Projective for Weierstrass

1986 Chudnovsky–Chudnovsky:
Speed up ADD by switching from
$(X/Z^2, Y/Z^3)$ to $(X/Z, Y/Z)$.
$7\mathbf{M} + 3\mathbf{S}$ for DBL if $a = -3$.
$12\mathbf{M} + 2\mathbf{S}$ for ADD.
$12\mathbf{M} + 2\mathbf{S}$ for reADD.

Option has been mostly ignored:
DBL dominates in ECDH etc.
But ADD dominates in
some applications: e.g.,
batch signature verification.

Montgomery curves

1987 Montgomery:

Use $by^2 = x^3 + ax^2 + x$.
Choose small $(a+2)/4$.

$$2(x_2, y_2) = (x_4, y_4)$$
$$\Rightarrow x_4 = \frac{(x_2^2 - 1)^2}{4x_2(x_2^2 + ax_2 + 1)}.$$

$$(x_3, y_3) - (x_2, y_2) = (x_1, y_1),$$
$$(x_3, y_3) + (x_2, y_2) = (x_5, y_5)$$
$$\Rightarrow x_5 = \frac{(x_2x_3 - 1)^2}{x_1(x_2 - x_3)^2}.$$

Represent (x, y)
as $(X{:}Z)$ satisfying $x = X/Z$.

$B = (X_2 + Z_2)^2$,
$C = (X_2 - Z_2)^2$,
$D = B - C$, $X_4 = B \cdot C$,
$Z_4 = D \cdot (C + D(a + 2)/4) \Rightarrow$
$2(X_2{:}Z_2) = (X_4{:}Z_4)$.

$(X_3{:}Z_3) - (X_2{:}Z_2) = (X_1{:}Z_1)$,
$E = (X_3 - Z_3) \cdot (X_2 + Z_2)$,
$F = (X_3 + Z_3) \cdot (X_2 - Z_2)$,
$X_5 = Z_1 \cdot (E + F)^2$,
$Z_5 = X_1 \cdot (E - F)^2 \Rightarrow$
$(X_3{:}Z_3) + (X_2{:}Z_2) = (X_5{:}Z_5)$.

This representation
does not allow ADD but it allows
DADD, "differential addition":
$Q, R, Q - R \mapsto Q + R$.

e.g. $2P, P, P \mapsto 3P$.
e.g. $3P, 2P, P \mapsto 5P$.
e.g. $6P, 5P, P \mapsto 11P$.

$2\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for DBL.
$4\mathbf{M} + 2\mathbf{S}$ for DADD.
Save $1\mathbf{M}$ if $Z_1 = 1$.

Easily compute $n(X_1 : Z_1)$ using
$\approx \lg n$ DBL, $\approx \lg n$ DADD.
Almost as fast as Edwards nP.
Relatively slow for $mP + nQ$ etc.

Doubling-oriented curves

2006 Doche–Icart–Kohel:

Use $y^2 = x^3 + ax^2 + 16ax$.
Choose small a.

Use $(X : Y : Z : Z^2)$
to represent $(X/Z, Y/Z^2)$.

$3\mathbf{M} + 4\mathbf{S} + 2\mathbf{D}$ for DBL.
How? Factor DBL as $\hat{\varphi}(\varphi)$
where φ is a 2-isogeny.

2007 Bernstein–Lange:
$2\mathbf{M} + 5\mathbf{S} + 2\mathbf{D}$ for DBL
on the same curves.

$12\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for ADD.
Slower ADD than other systems,
typically outweighing benefit
of the very fast DBL.

But isogenies are useful.
Example, 2005 Gaudry:
fast DBL+DADD on Jacobians of
genus-2 hyperelliptic curves,
using similar factorization.

Tricky but potentially helpful:
tripling-oriented curves
(see 2006 Doche–Icart–Kohel),
double-base chains, ...

Hessian curves

Credited to Sylvester
by 1986 Chudnovsky–Chudnovsky:

$(X : Y : Z)$ represent $(X/Z, Y/Z)$
on $x^3 + y^3 + 1 = 3dxy$.

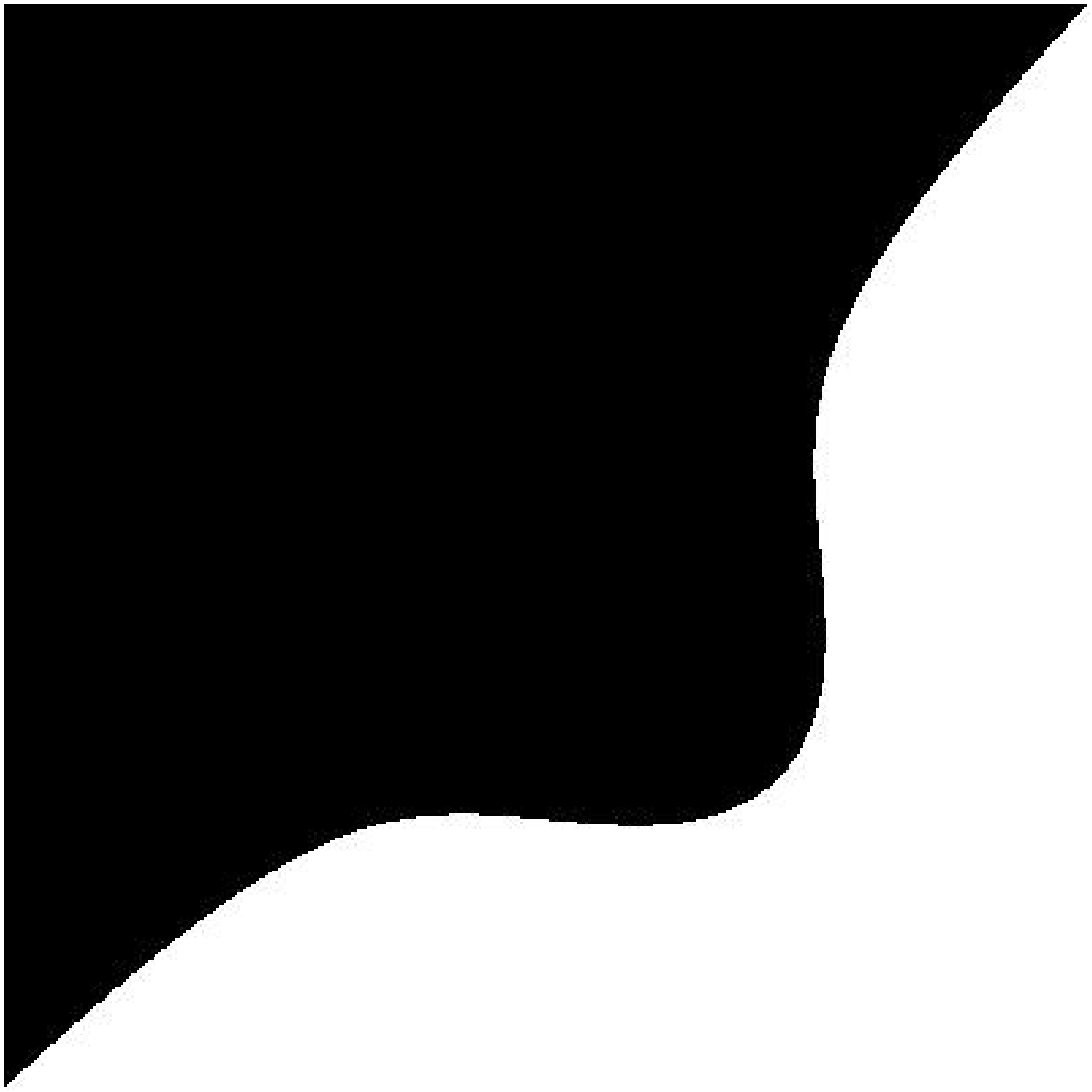
12**M** for ADD:
$X_3 = Y_1X_2 \cdot Y_1Z_2 - Z_1Y_2 \cdot X_1Y_2$,
$Y_3 = X_1Z_2 \cdot X_1Y_2 - Y_1X_2 \cdot Z_1X_2$,
$Z_3 = Z_1Y_2 \cdot Z_1X_2 - X_1Z_2 \cdot Y_1Z_2$.

6**M** + 3**S** for DBL.

2001 Joye–Quisquater:
$2(X_1 : Y_1 : Z_1) =$
$(Z_1 : X_1 : Y_1) + (Y_1 : Z_1 : X_1)$
so can use ADD to double.

"Unified addition formulas,"
helpful against side channels.
But need to permute inputs.
2009 Bernstein–Kohel–Lange:
Easily avoid permutation!

2008 Hisil–Wong–Carter–Dawson:
$(X : Y : Z : X^2 : Y^2 : Z^2$
$\quad : 2XY : 2XZ : 2YZ).$
$6\mathbf{M} + 6\mathbf{S}$ for ADD.
$3\mathbf{M} + 6\mathbf{S}$ for DBL.



$$x^3 - y^3 + 1 = 0.3xy$$


The Hessian-ray: uniform
but
not strongly so

Jacobi intersections

1986 Chudnovsky–Chudnovsky:

$(S : C : D : Z)$ represent
$(S/Z, C/Z, D/Z)$ on
$s^2 + c^2 = 1$, $as^2 + d^2 = 1$.

$14\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for ADD.
"Tremendous advantage"
of being strongly unified.

$5\mathbf{M} + 3\mathbf{S}$ for DBL.
"Perhaps (?) ... the most
efficient duplication formulas
which do not depend on the
coefficients of an elliptic curve."

2001 Liardet–Smart:

$13\mathbf{M} + 2\mathbf{S} + 1\mathbf{D}$ for ADD.

$4\mathbf{M} + 3\mathbf{S}$ for DBL.

2007 Bernstein–Lange:

$3\mathbf{M} + 4\mathbf{S}$ for DBL.

2008 Hisil–Wong–Carter–Dawson:

$13\mathbf{M} + 1\mathbf{S} + 2\mathbf{D}$ for ADD.

$2\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for DBL.

Also $(S : C : D : Z : SC : DZ)$:

$11\mathbf{M} + 1\mathbf{S} + 2\mathbf{D}$ for ADD.

$2\mathbf{M} + 5\mathbf{S} + 1\mathbf{D}$ for DBL.

Jacobi quartics

$(X{:}Y{:}Z)$ represent $(X/Z, Y/Z^2)$
on $y^2 = x^4 + 2ax^2 + 1$.

1986 Chudnovsky–Chudnovsky:
$3\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$ for DBL.
Slow ADD.

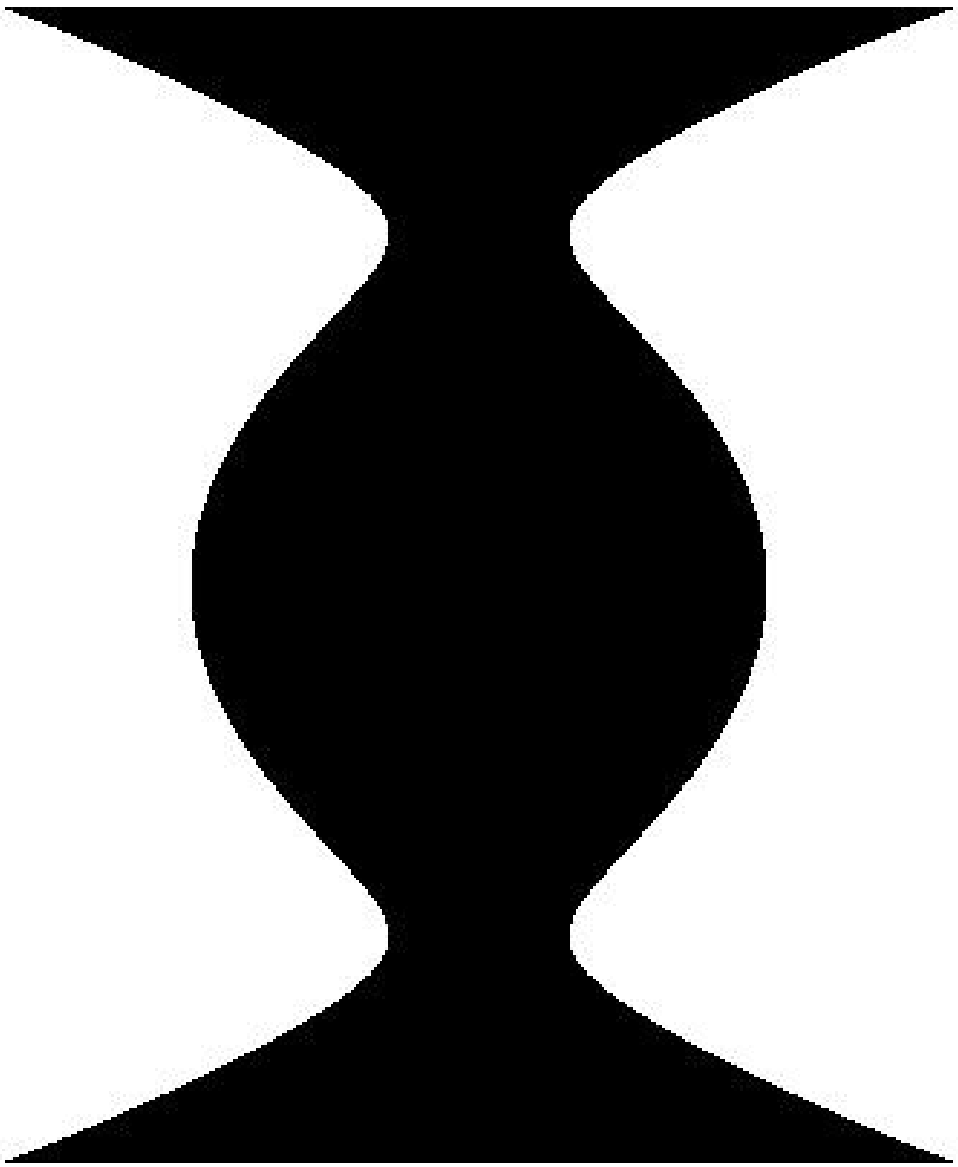
2002 Billet–Joye:
New choice of neutral element.
$10\mathbf{M} + 3\mathbf{S} + 1\mathbf{D}$ for ADD,
strongly unified.

2007 Bernstein–Lange:
$1\mathbf{M} + 9\mathbf{S} + 1\mathbf{D}$ for DBL.

2007 Hisil–Carter–Dawson:
$2\mathbf{M} + 6\mathbf{S} + 2\mathbf{D}$ for DBL.

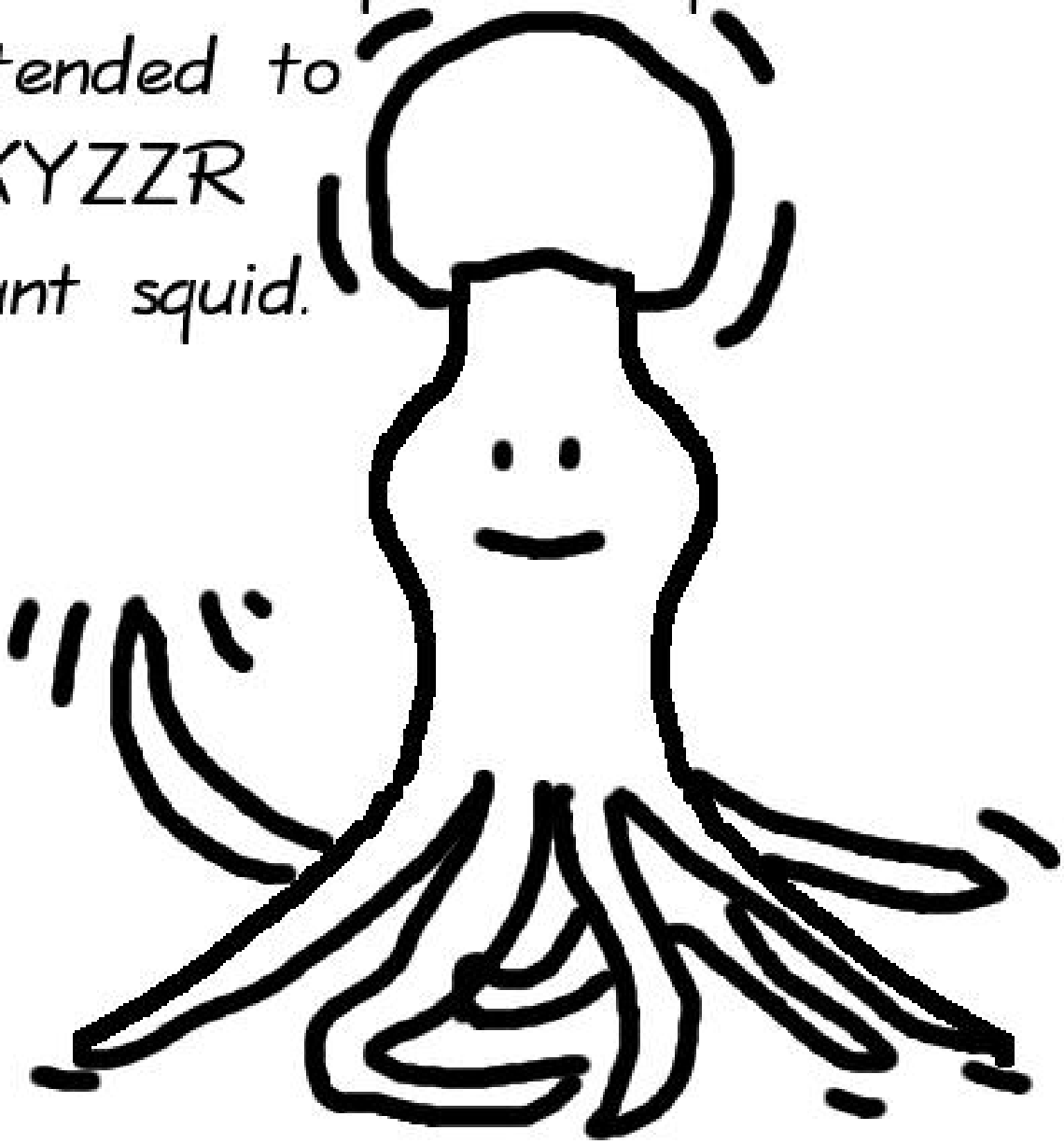
2007 Feng–Wu:
$2\mathbf{M} + 6\mathbf{S} + 1\mathbf{D}$ for DBL.
$1\mathbf{M} + 7\mathbf{S} + 3\mathbf{D}$ for DBL
on curves chosen with $a^2 + c^2 = 1$.

More speedups: 2007 Duquesne,
2007 Hisil–Carter–Dawson,
2008 Hisil–Wong–Carter–Dawson:
use $(X : Y : Z : X^2 : Z^2)$
or $(X : Y : Z : X^2 : Z^2 : 2XZ)$.
Can combine with Feng–Wu.
Competitive with Edwards!

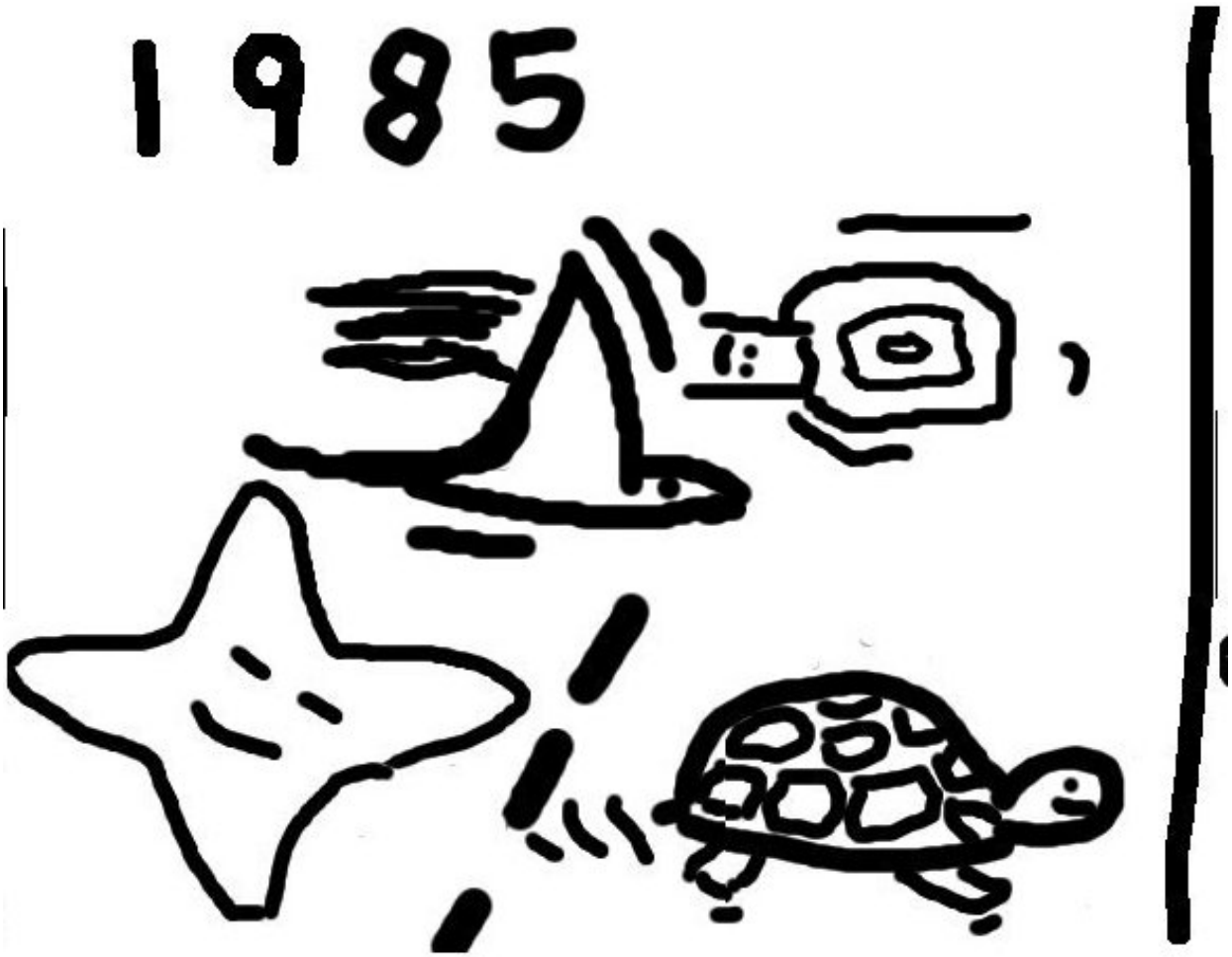


$$x^2 = y^4 - 1.9y^2 + 1$$

The Jacobi-quartic squid: can be extended to XXYZZR giant squid.

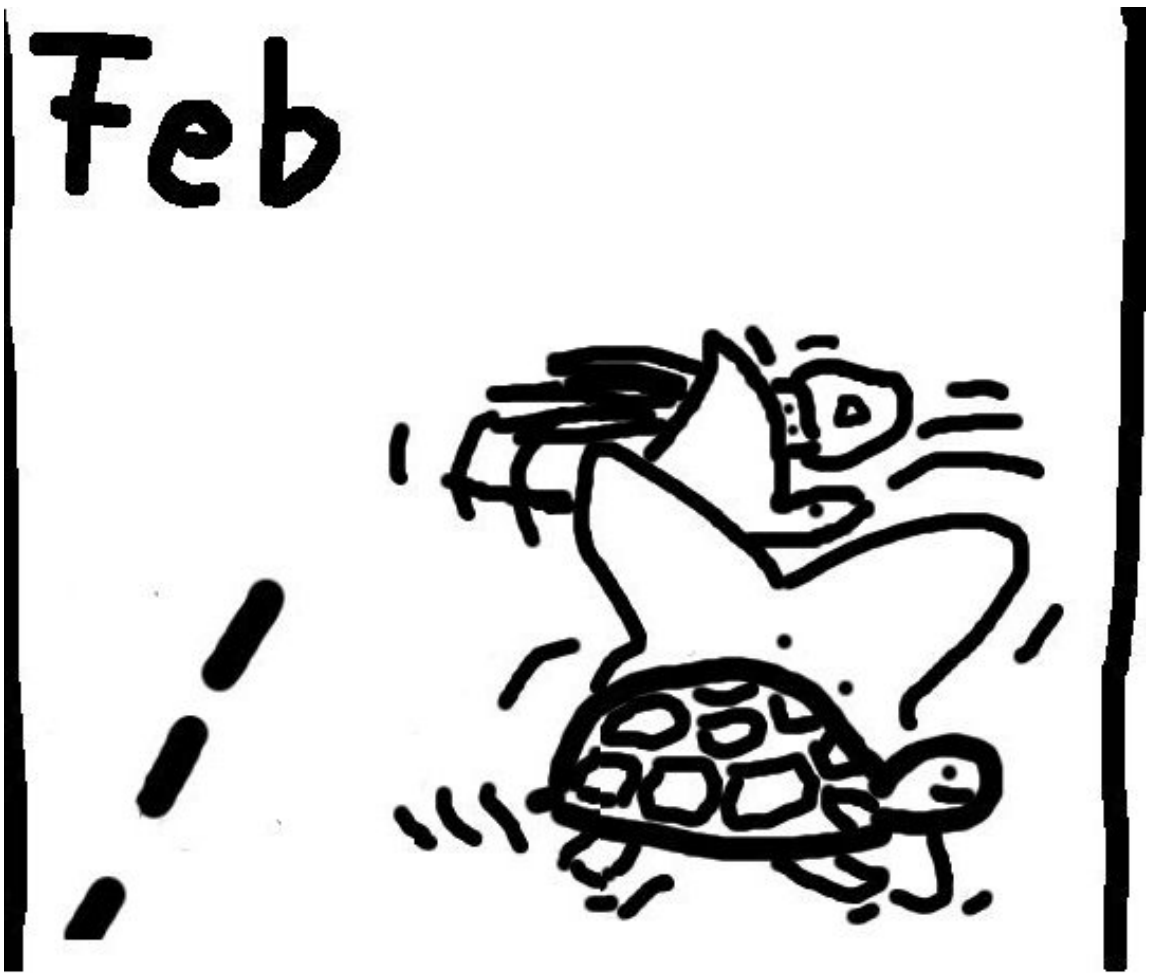



START

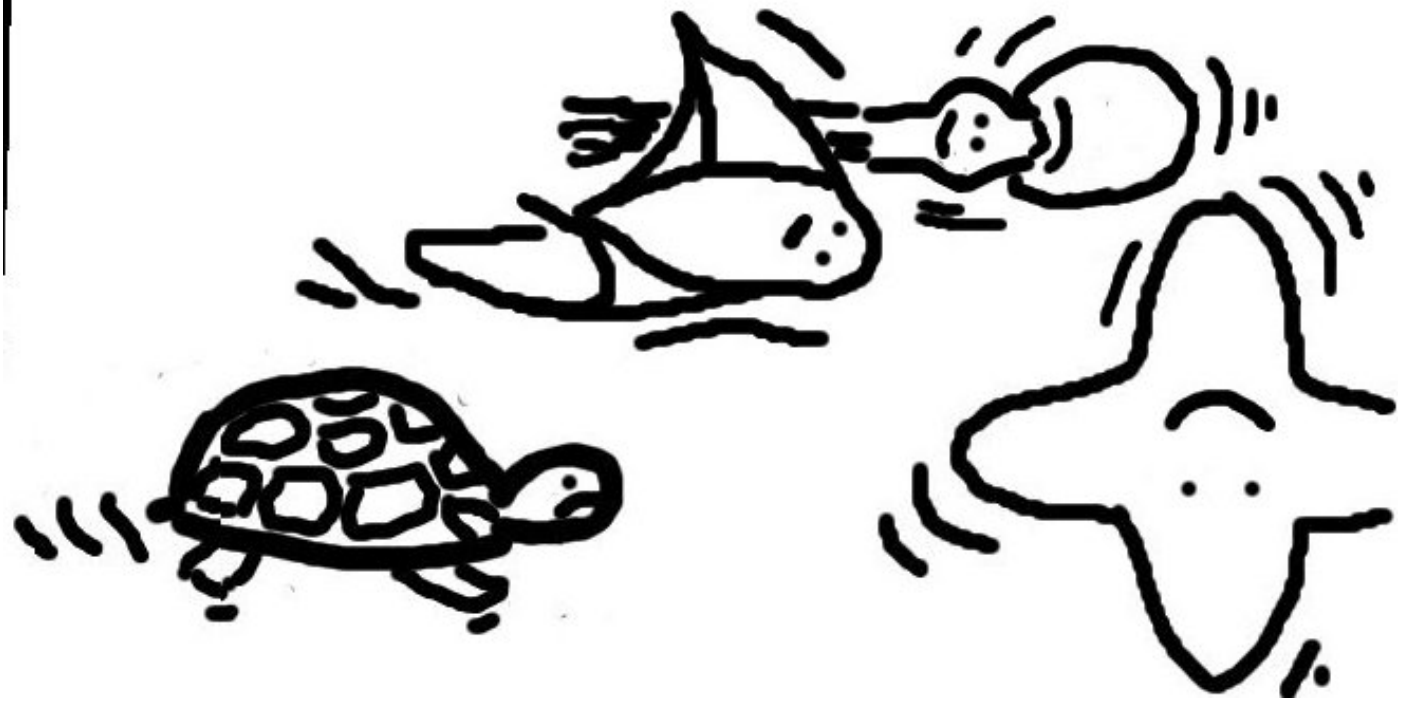

1985

2007-Jan

Feb



Mar



More addition formulas

Explicit-Formulas Database:
hyperelliptic.org/EFD

EFD has 581 computer-verified formulas and operation counts for ADD, DBL, etc.
in 51 representations
on 13 shapes of elliptic curves.

Not yet handled by computer:
generality of curve shapes
(e.g., Hessian order $\in 3\mathbf{Z}$);
complete addition algorithms
(e.g., checking for ∞).