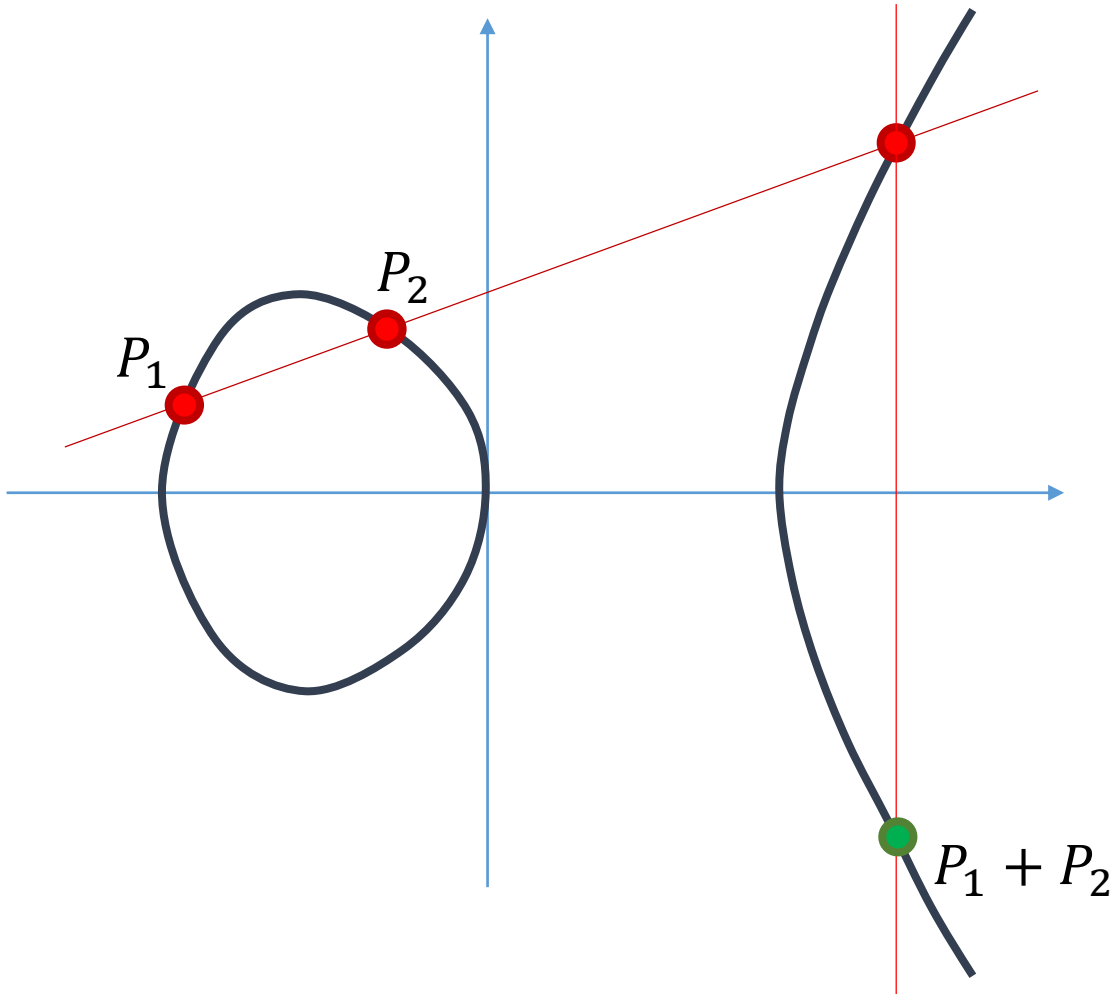
Elliptic curve arithmetic



ECC school, Nijmegen, 9-11 November 2017

Wouter Castryck

KU LEUVEN

COSIC

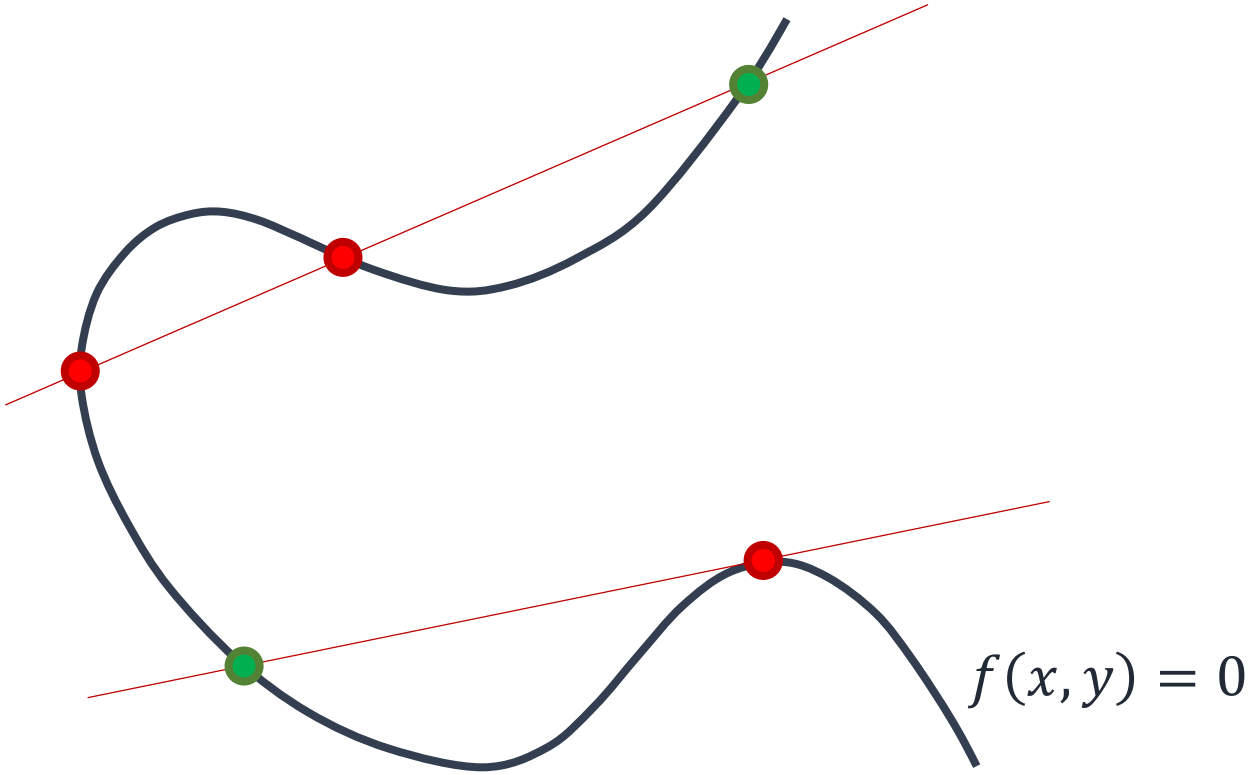
Tangent-chord arithmetic on cubic curves

Introduction

Consequence of **Bézout**'s theorem: on a cubic curve

$$C : f(x,y) = \sum_{i+j=3} a_{ij} x^i y^j = 0,$$

new points can be constructed from known points using tangents and chords.



Pierre de Fermat

Isaac Newton

This principle was already known to 17th century natives like **Fermat** and **Newton**.

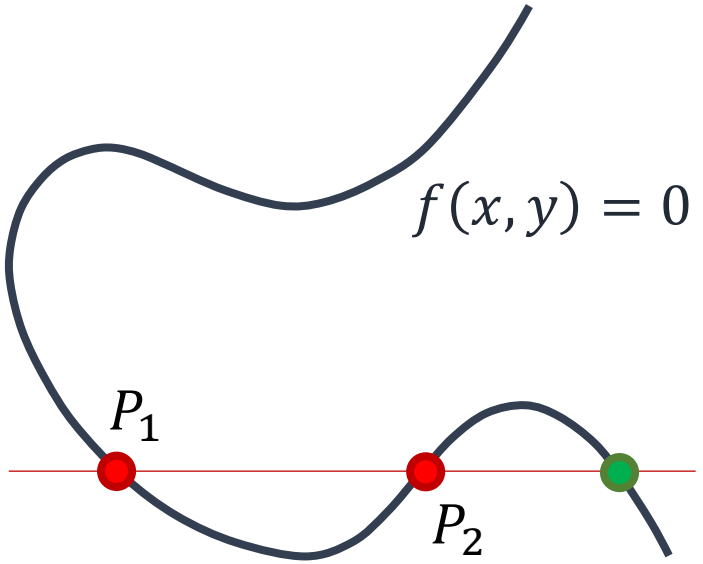
Introduction

This construction was known to **respect the base field**.

This means: if $f(x, y) \in k[x, y]$ with k some field, and one starts from points having coordinates in k, then new points obtained through the tangent-chord method also have coordinates in k.

Informal reason:

Consider two points on the x-axis $P_1 = (a, 0)$ and $P_2 = (b, 0)$.
Then the "chord" is $y = 0$.



The intersection is computed by $f(x, 0) = (x - a) \cdot (x - b) \cdot \text{linear factor}$

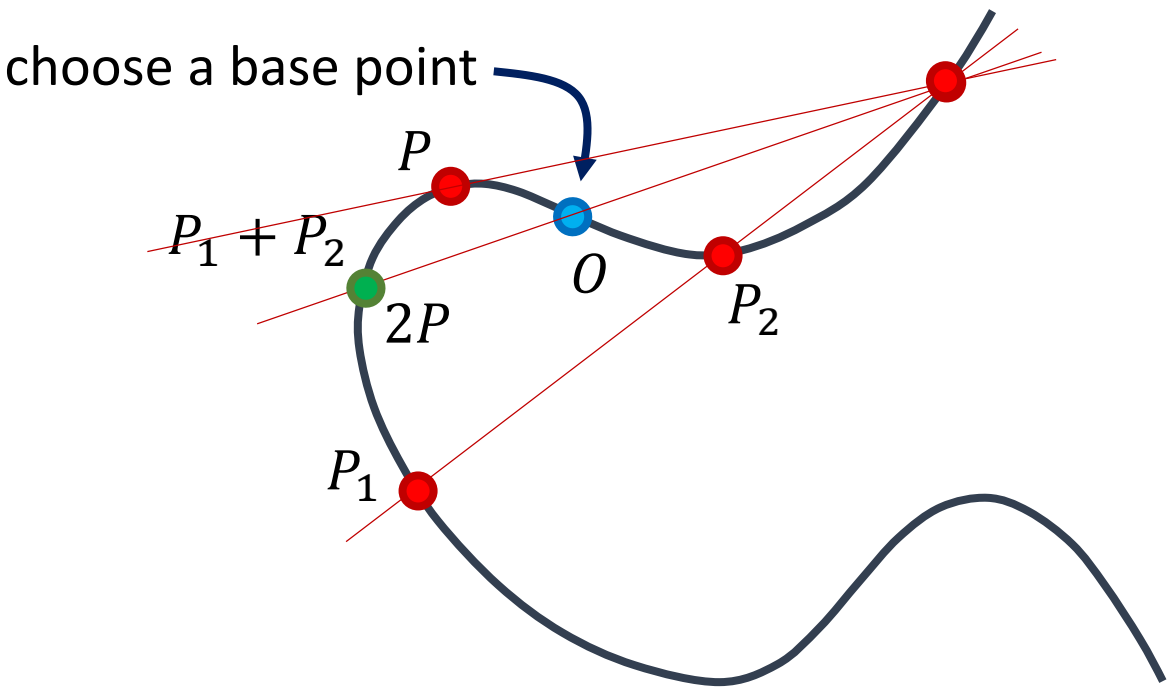
always has a root over k!

Introduction

Thus: tangents and chords give some sort of composition law on the set of k-rational points of a cubic curve.

Later it was realized that by adding in a second step, this gives the curve an **abelian group** structure!

only after **an incredible historical detour** which took more than 200 years…



First formalized by **Poincaré** in 1901.

Henri Poincaré

commutativity:

$$P_1 + P_2 = P_2 + P_1$$

associativity:

$$(P_1 + P_2) + P_3 = P_1 + (P_2 + P_3)$$

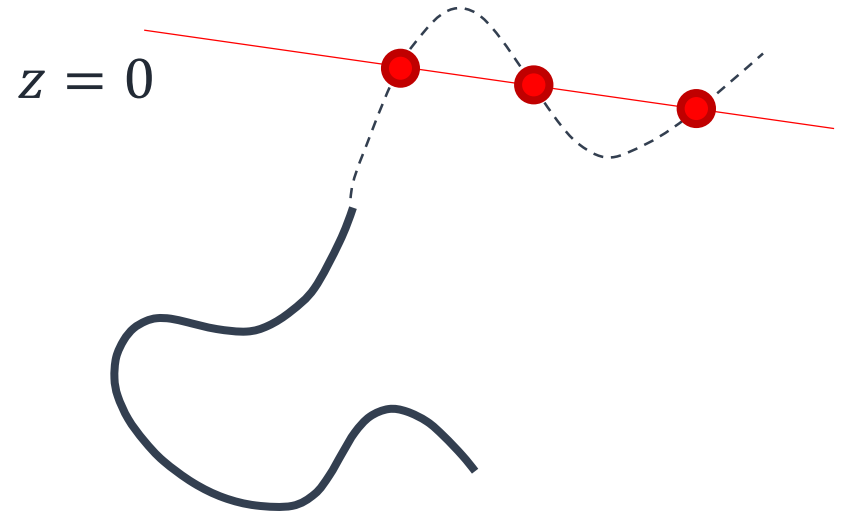
neutral element:

$$P + O = P$$

inverse element:

$$\exists(-P) : P + (-P) = O$$

Introduction



Conditions for this to work:

1) One should work **projectively** (as opposed to affinely):

Homogenize

$f(x, y) = \sum_{i+j=3} a_{ij} x^i y^j$ to $F(x, y, z) = \sum_{i+j=3} a_{ij} x^i y^j z^{3-i-j}$

and consider points $(x : y : z) \neq (0 : 0 : 0)$, up to scaling.

Two types of points:

affine points

$z \neq 0$: the point is of the form $(x : y : 1)$

But then (x, y) is an affine point!

points at infinity

$z = 0$: points of the form $(x : y : 0)$ up to scaling.

(Up to three such points.)

Introduction

Conditions for this to work:

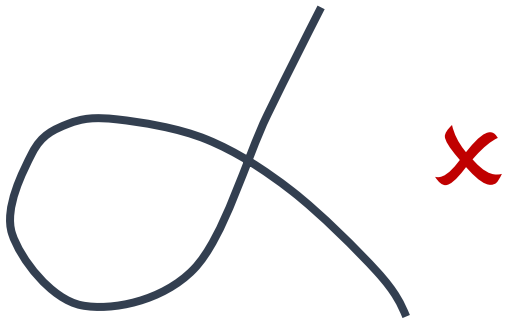
2) The curve should be **smooth**, meaning that

$$f = \frac{\partial f}{\partial x} = \frac{\partial f}{\partial y} = \frac{\partial f}{\partial z} = 0$$

has no solutions.

This ensures that every point P has a well-defined tangent line

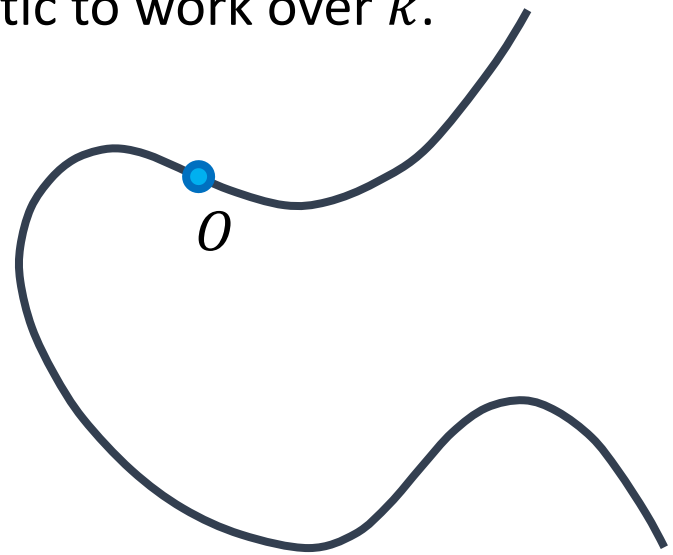
$$T : \frac{\partial f}{\partial x}(P) \cdot x + \frac{\partial f}{\partial y}(P) \cdot y + \frac{\partial f}{\partial z}(P) \cdot z = 0.$$



Introduction

Conditions for this to work:

3) O should have coordinates in k, in order for the arithmetic to work over k.



Definition: *an elliptic curve over k is a smooth projective cubic curve E/k equipped with a k-rational base point O.*

(Caution: there exist more general and less general definitions.)

Under these assumptions we have as wanted:

Tangent-chord arithmetic turns E into an abelian group with neutral element O.

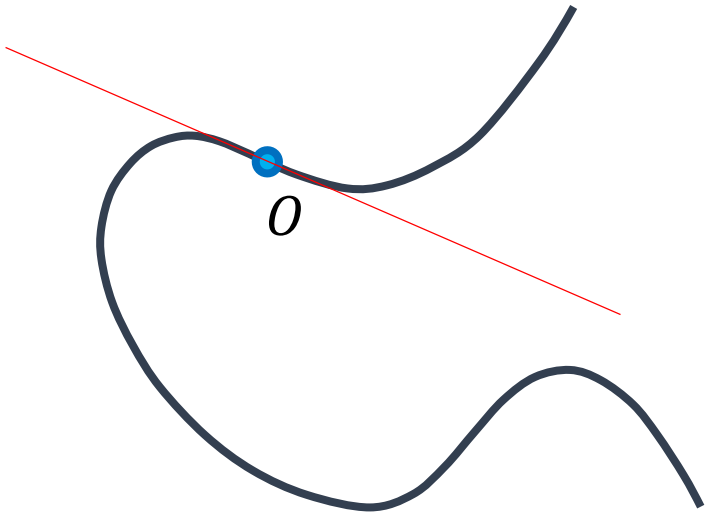
The set of k-rational points $E(k)$ form a subgroup.

Exercises

1) Describe geometrically what it means to **invert** a point P, i.e. to find a point $-P$ such that

$$P + (-P) = O.$$

2) Why does this construction simplify considerably if O is a **flex** (= point at which its tangent line meets the curve triply)?



3) If O is a flex then

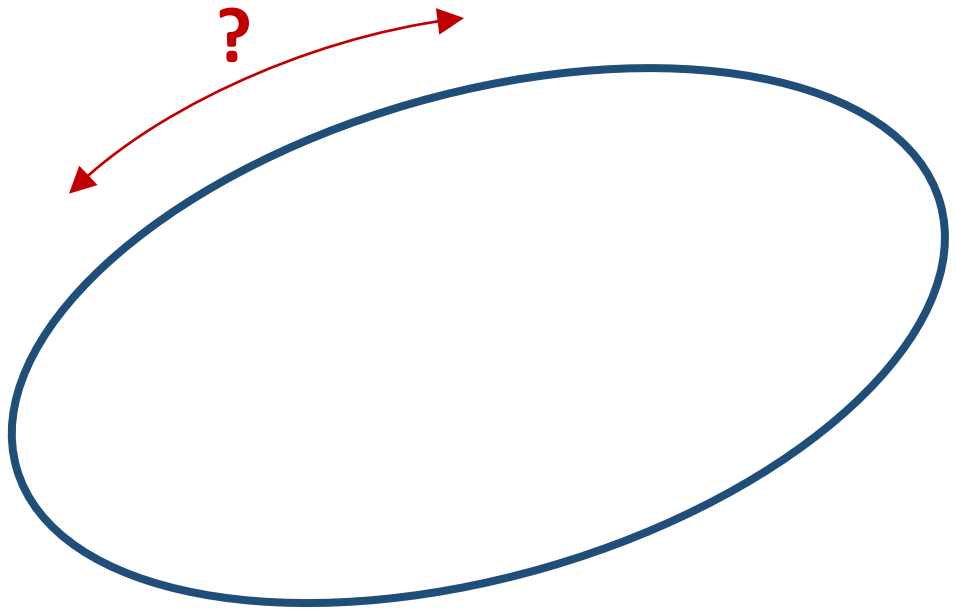
$3P := P + P + P = O$ if and only if P is a flex.

Explain why.

On the terminology “elliptic curves”

On the terminology

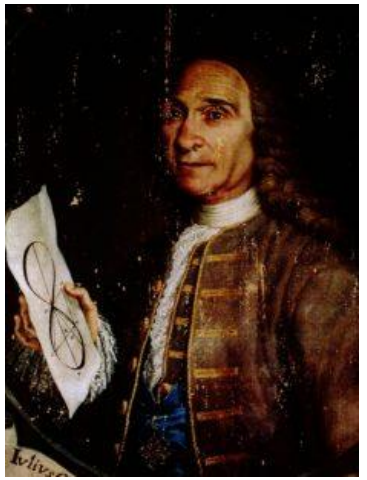
In the 18th century, unrelated to all this, **Fagnano** and **Euler** revisited the unsolved problem of determining the circumference of an ellipse.



$$\int_0^t \frac{\sqrt{1-2x^2}}{\sqrt{1-x^2}}\,dx$$

$$\int_0^t \frac{dx}{\sqrt{x^3+x+1}}$$

$$\int_0^t \frac{dx}{\sqrt{1-x^4}}$$



Giulio Fagnano

Leonhard Euler

They got stuck on difficult integrals, now called *elliptic integrals*.

On the terminology

In the 19th century **Abel** and **Jacobi** studied the inverse functions of elliptic integrals.

$$s = \int_0^t \frac{\sqrt{1-2x^2}}{\sqrt{1-x^2}}\,dx$$

$$t = f(s)\ ?$$

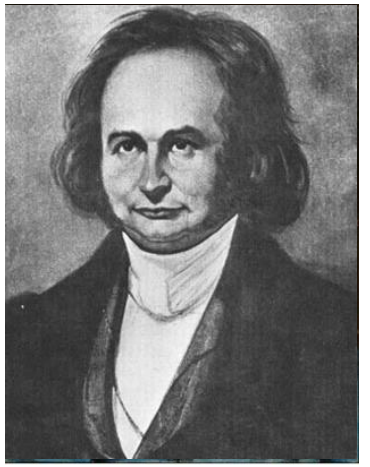
$$s = \int_0^t \frac{dx}{\sqrt{x^3+x+1}}$$

$$s = \int_0^t \frac{dx}{\sqrt{1-x^4}}$$

Niels H. Abel

When viewed as complex functions, they observed doubly periodic behaviour: there exist $\omega_1, \omega_2 \in \mathbf{C}$ such that

$$f(z + \lambda_1\omega_1 + \lambda_2\omega_2) = f(z) \quad \text{for all } \lambda_1, \lambda_2 \in \mathbf{Z}.$$



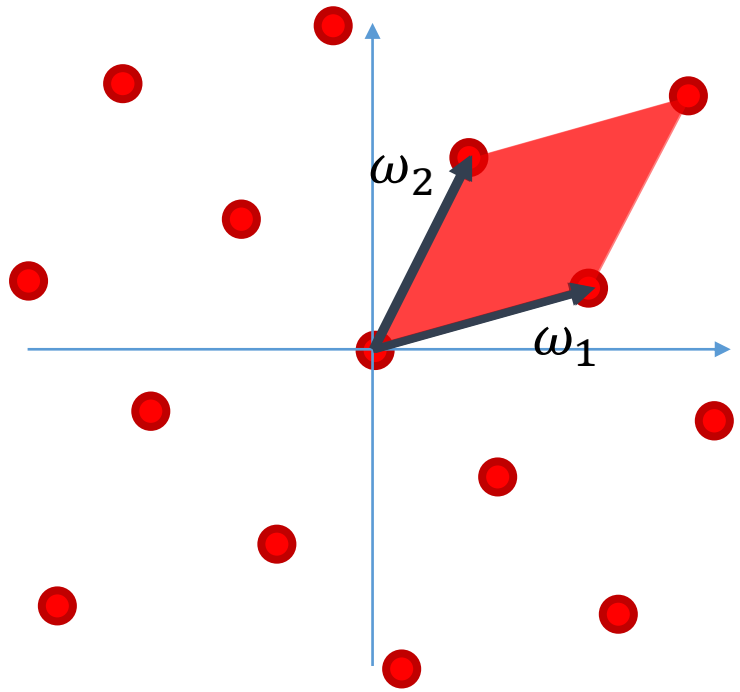
Carl G. Jacobi

Compare to: $\sin(x + \lambda \cdot 2k\pi) = \sin(x)$ for all $\lambda \in \mathbf{Z}$, etc.

Such generalized trigonometric functions became known as *elliptic functions*.

On the terminology

In other words: elliptic functions on $\mathbf{C}$ are well-defined modulo $\mathbf{Z}\omega_1 + \mathbf{Z}\omega_2$.

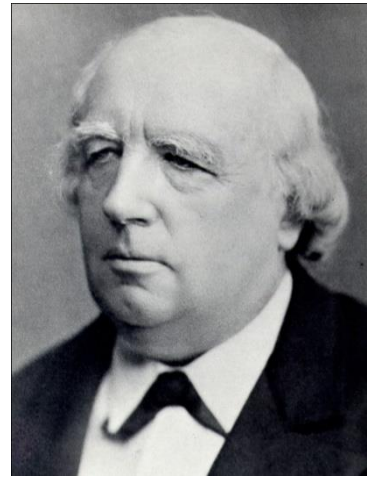


Mid 19th century **Weierstrass** classified all elliptic functions for any given ω_1, ω_2, and used this to define a biholomorphism

$$\mathbf{C}/(\mathbf{Z}\omega_1 + \mathbf{Z}\omega_2) \to E : z \mapsto (\wp(z), \wp'(z))$$

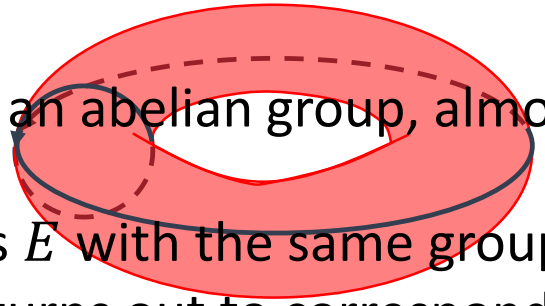
to a certain algebraic curve E…

… which he called an *elliptic* curve!



Karl Weierstrass

Note that $\mathbf{C}/(\mathbf{Z}\omega_1 + \mathbf{Z}\omega_2)$ is an abelian group, almost by definition.



The biholomorphism endows E with the same group structure…

… where it turns out to correspond to tangent-chord arithmetic!

Weierstrass curves and their arithmetic

Weierstrass curves

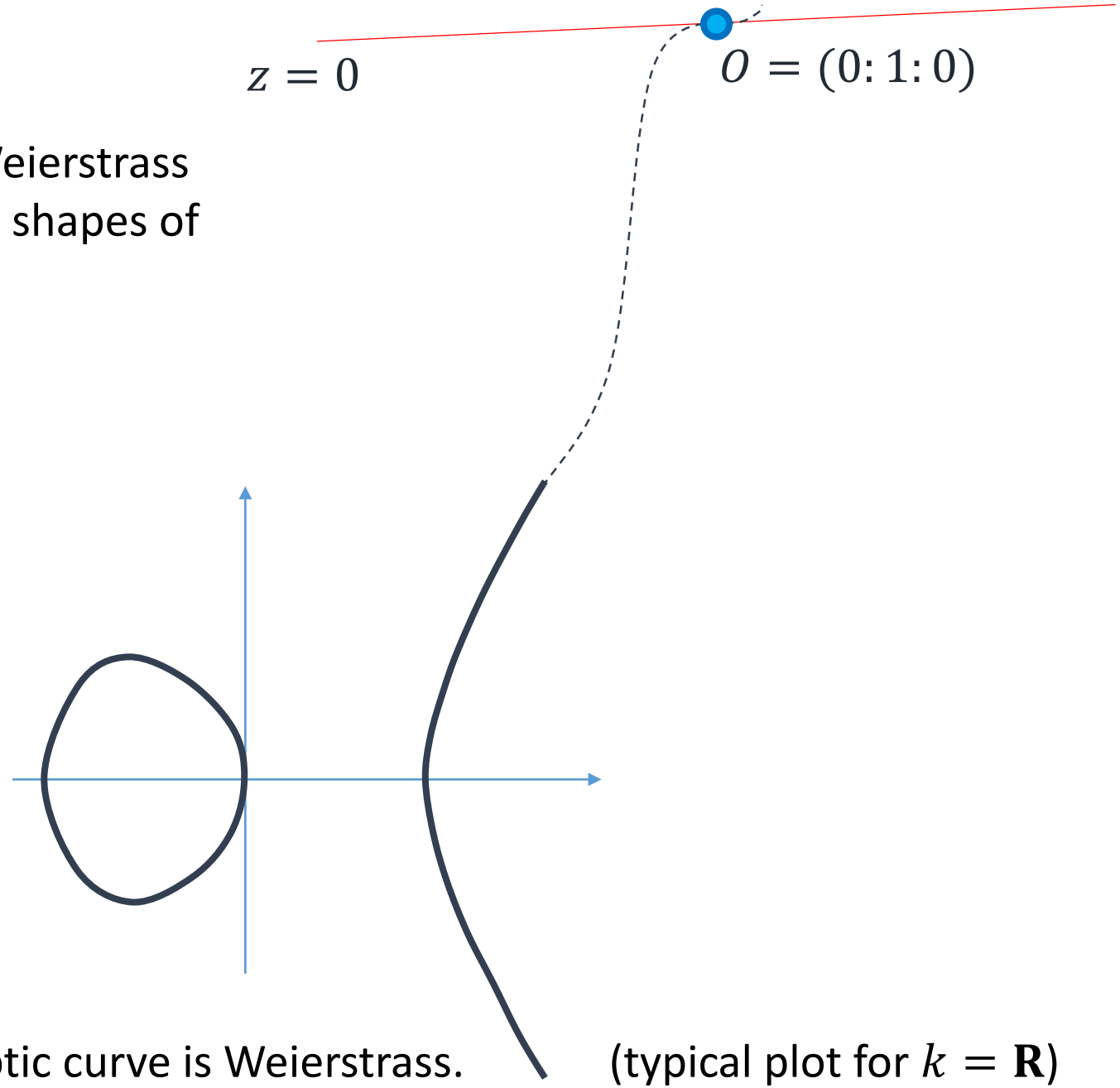
The concrete type of elliptic curves found by Weierstrass now carry his name. They are the most famous shapes of elliptic curves.

Assume $\mathrm{char}\ k \neq 2,3$.

Definition: *a Weierstrass elliptic curve is defined by*

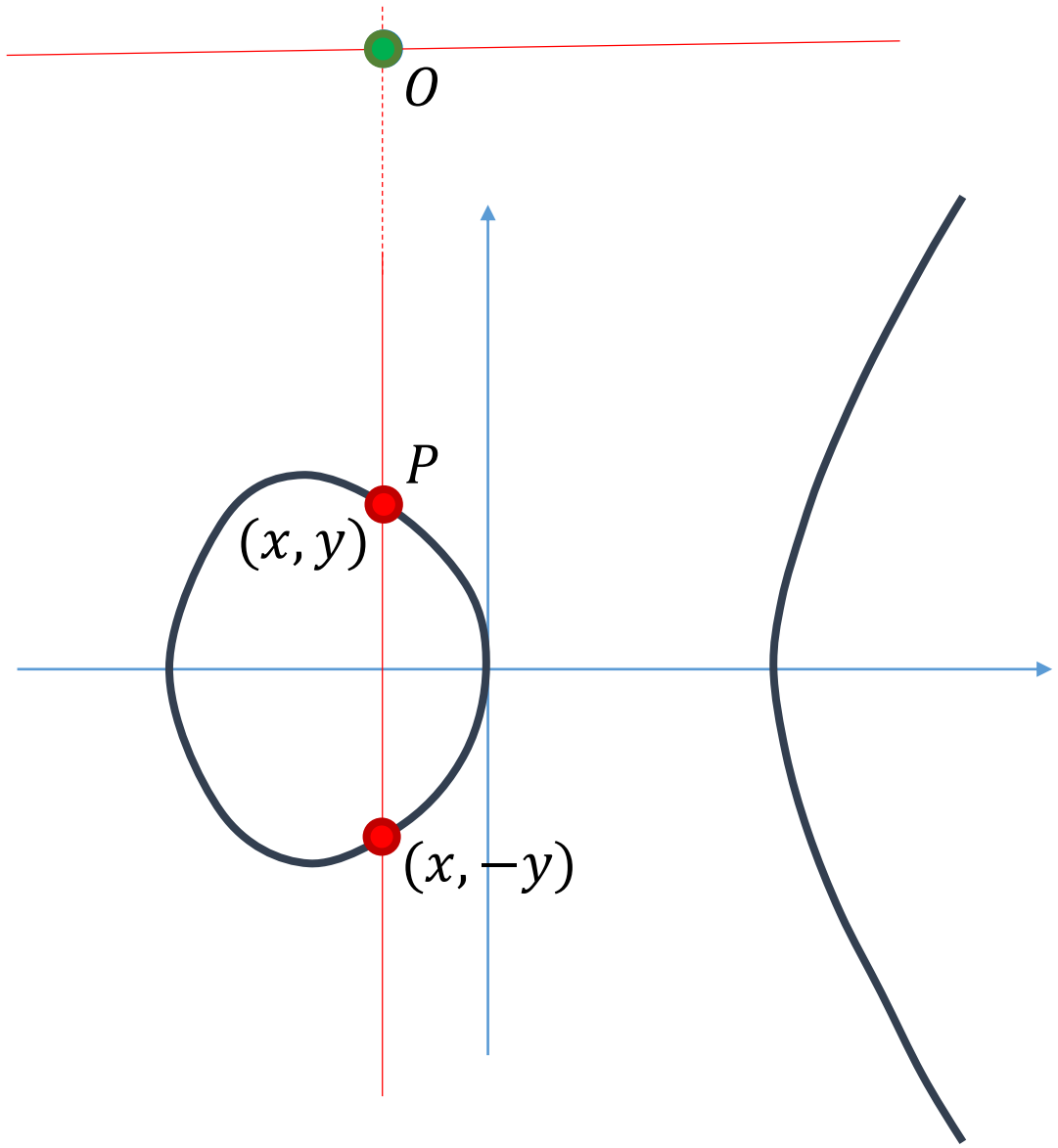
$$y^2z = x^3 + Axz^2 + Bz^3$$

where $A, B \in k$ satisfy $4A^3 + 27B^2 \neq 0$. The base point O is the unique point at infinity.



Can be shown: up to "isomorphism" every elliptic curve is Weierstrass. (typical plot for $k = \mathbf{R}$)

Weierstrass curves



Note:

1) the lines through $O = (0:1:0)$ are the **vertical lines** (except for the line at infinity $z = 0$).

2) The equation $y^2 = x^3 + Ax + B$ is symmetric in y.

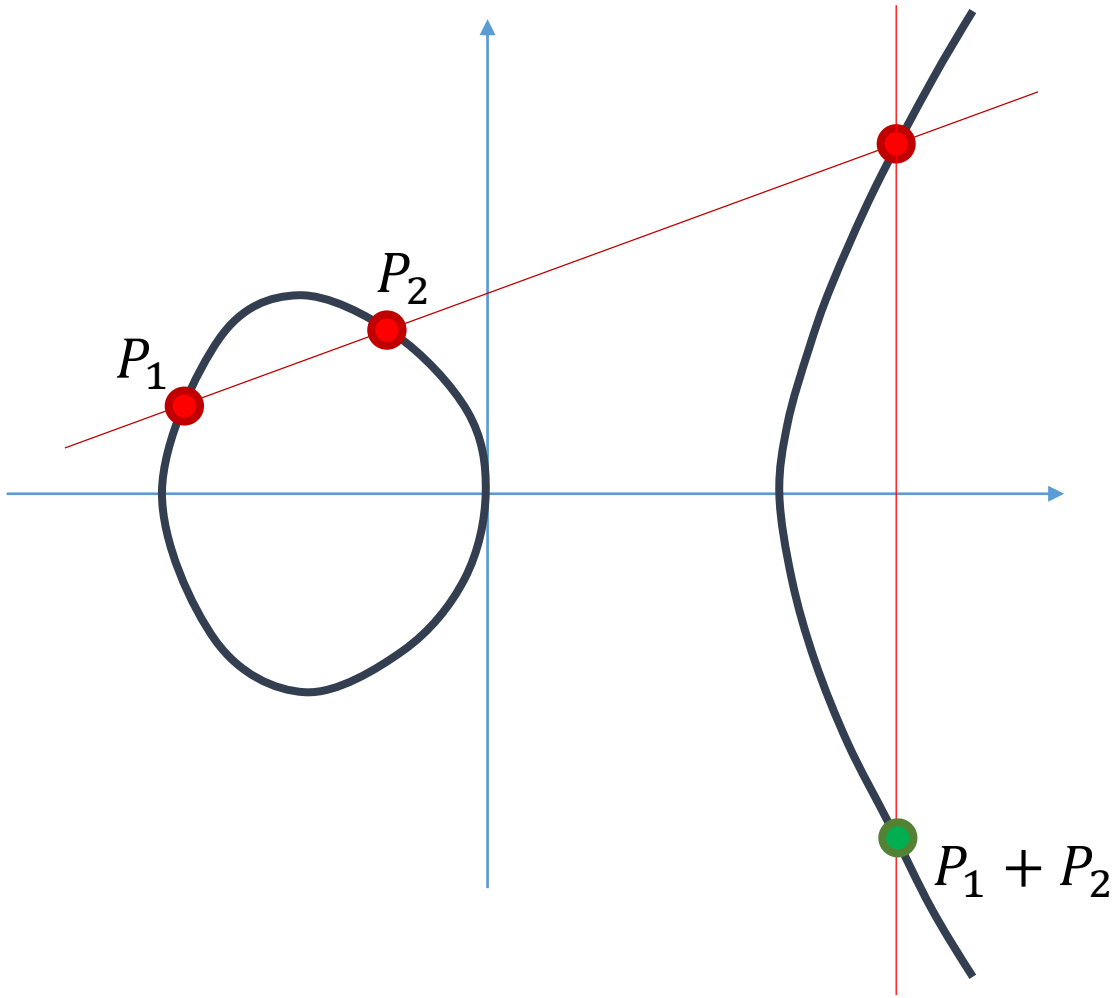
This gives a first feature: inverting a point on a Weierstrass curve is super easy!

Indeed: if $P = (x, y)$ is an affine point then

$$-P = (x, -y).$$

Weierstrass curves

What about point addition?



Write $P_1 + P_2 = (x_3, y_3)$.

Line through $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ is

$$y - y_1 = \lambda(x - x_1) \quad \text{where} \quad \lambda = \frac{y_2 - y_1}{x_2 - x_1}.$$

Substituting $y \leftarrow y_1 + \lambda(x - x_1)$ in the curve equation $x^3 + Ax + B - y^2 = 0$:

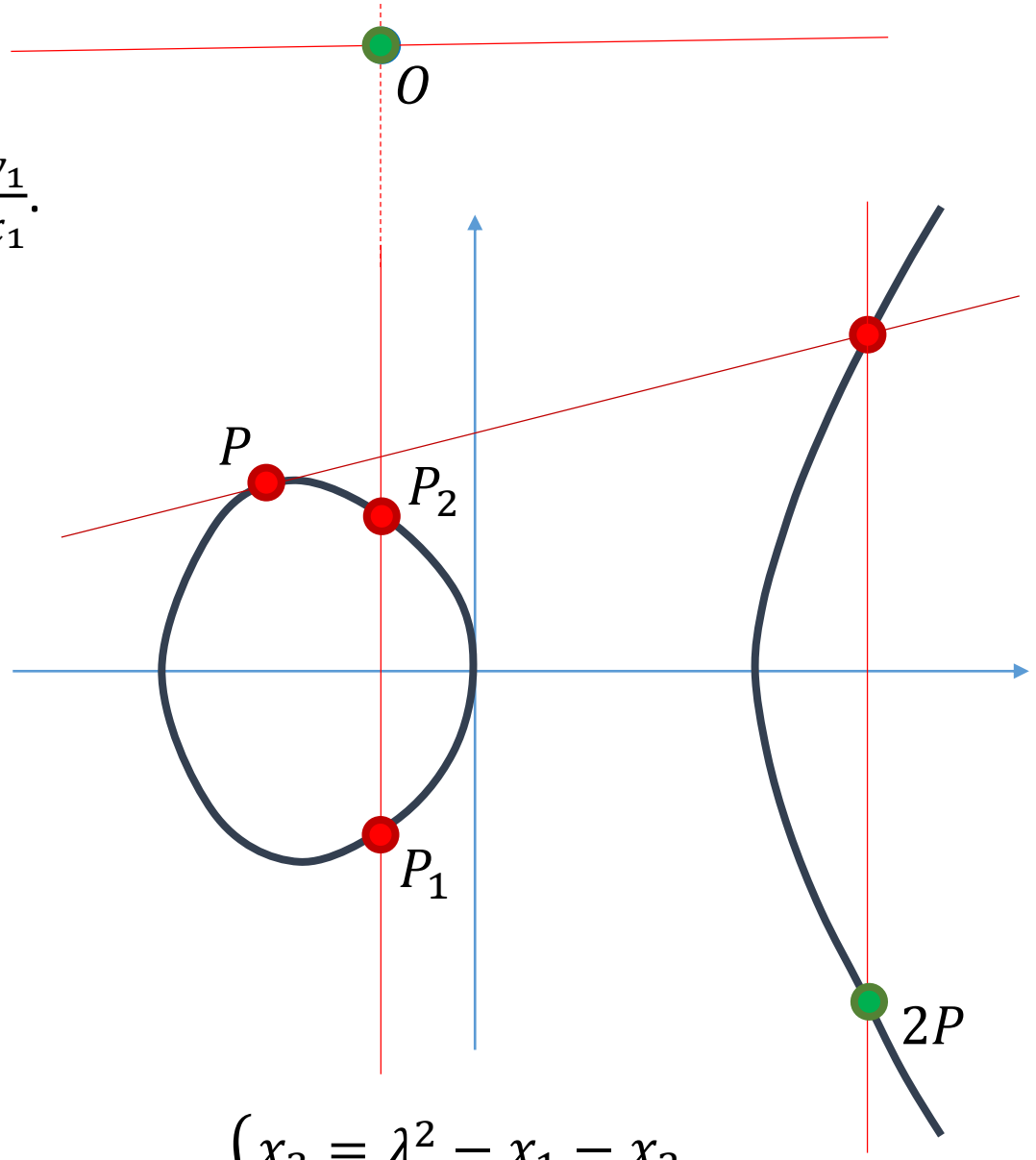
$$x^3 + Ax + B - \left(y_1 + \lambda(x - x_1)\right)^2 = 0.$$

So, sum of the roots is λ^2. But x_1, x_2 are roots!

We find: $\begin{cases} x_3 = \lambda^2 - x_1 - x_2 \\ y_3 = -y_1 - \lambda(x_3 - x_1) \end{cases}$

Weierstrass curves

where $\lambda = \frac{y_2 - y_1}{x_2 - x_1}$.



But what if $x_1 = x_2$?

Two cases:

Either $y_1 = y_2 \neq 0$, i.e. $P_1 = P_2 = P$.

In this case we need to replace λ by

$$\lambda = \frac{3x_1^2 + 2Ax_1}{2y_1}.$$

Or $y_1 = -y_2$, in which case $P_1 + P_2 = O$.

We find: $\begin{cases} x_3 = \lambda^2 - x_1 - x_2 \\ y_3 = -y_1 - \lambda(x_3 - x_1) \end{cases}$

Conclusion: formulas for computing on a Weierstrass curve are not too bad, but case distinctive.

More efficient elliptic curve arithmetic?

The Weierstrass addition formulas are reasonably good for several purposes…

… but can they be boosted? Huge amount of activity starting in the 1980's.

Victor Miller

One reason:

Koblitz and Miller's suggestion to use elliptic curves in crypto!

agree on $E/\mathbf{F}_q$ and $P \in E(\mathbf{F}_q)$

chooses secret $a \in \mathbf{Z}$

receives bP

computes $a(bP) = abP$

chooses secret $b \in \mathbf{Z}$

receives aP

computes $b(aP) = abP$

(Example: Diffie-Hellman key exchange.)

Initial reason:

Lenstra's elliptic curve method (ECM) for integer factorization.

Neal Koblitz

Generic methods for efficient scalar multiplication

Efficient scalar multiplication

The most important operation in both
(discrete-log based) elliptic curve cryptography,
the elliptic curve method for integer factorization,
is **scalar multiplication**: given a point P and a positive integer a, compute

$$aP := \underbrace{P + P + \cdots + P}_{a \text{ times.}}$$

Note: adding P consecutively to itself $a - 1$ times is **not an option**!

in practice a consists of hundreds of bits!

Efficient scalar multiplication: double-and-add

Much better idea: **double-and-add**, walking through the binary expansion of a.

Toy example: replace the 15 additions in

$$16P = P + P + P + P + P + P + P + P + P + P + P + P + P + P + P + P$$

by the 4 doublings in

$$16P = 2\left(2(2(2P))\right).$$

General method:

$$a = 101100010 \ldots 0101 \qquad \mathbf{2\left(2\left(2(\cdots)\right)\right) + P}$$

double add

Exercise: verify that this computes aP using $O(\log a)$ additions or doublings, as opposed to $O(a)$. (Horner's rule, basically.)

Efficient scalar multiplication: double-and-add

Asymptotically this is as good as we can expect…

… but in practice, considerable speed-ups over naive double-and-add are possible!

Example: double-and-add computes $15P$ as

$$P, 2P, 3P, 6P, 7P, 14P, 15P.$$

However it would have been more efficient to compute it as

$$P, 2P, 3P, 6P, 12P, 15P$$

Warning: finding the most optimal chain of additions and doublings to compute aP is a very difficult combinatorial problem.

We don't want to spend more time on it than on computing aP itself!

Efficient scalar multiplication: windowing

In double-and-add, processing a 0 (doubling) is less costly than processing a 1 (doubling and adding P). Is there a structural way of reducing the number of additions?

One idea to achieve this: **windowing**, which is the same as double-and-add, but we now process blocks (= windows) of w bits in one time.

Example with $w = 2$:

$$a = \underline{10}\,\underline{11}\,\underline{00}\,\underline{10} \ldots 0101$$

quadruple and add $3P$ add P

$$4\left(4\left(4\left(4P\right)+3P\right)\right)+P$$

Requires precomputation of $P, \ldots, 2^{w-1}P$ which grows exponentially with w.

Method can be spiced up by allowing the window to *slide* to the next window starting with a 1.

Efficient scalar multiplication: signed digits

Recall that on a Weierstrass elliptic curve, inverting a point is quasi cost-free:

$$-(x, y) = (x, -y).$$

Idea: use negative digits in the expansion, at the benefit of having more 0's.

The **non-adjacent form (NAF)** of an integer a is a base 2 expansion
-> with digits taken from $\{-1,0,1\}$
-> in which no two consecutive digits are non-zero.

Such an expansion always exists, is unique, and easy to find.

$a =$ 1 0 0 -1 0 -1 0 1 0 … 0 1 0 0 1

This method also comes in a windowing version (w-NAF).

Efficient scalar multiplication

Tons of variations to the foregoing ideas have been investigated and proposed.

Some examples (far from exhaustive!):

Work with respect to base 3 and use an expansion with digits $\in \{-1,0,1\}$.
(Requires a tripling formula.)

If P has a known finite order n, check if $a \pm \lambda n$ has better properties for some small $\lambda \in \mathbf{Z}$.

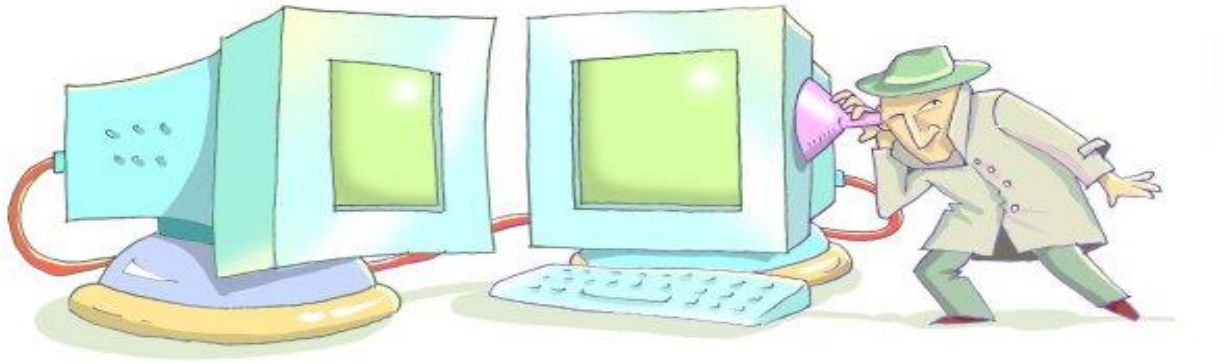
Multi-exponentiation: efficient methods for computing a $\mathbf{Z}$-linear combination $\sum_i a_i P_i$.

Exercise: find a smarter way to compute $aP + bQ$ than first computing aP, bQ separately.

Caution with double-and-add and its variants

When working through the digits of the scalar

$$a = 1101001101011000 10 \ldots 1110,$$



an attacker might notice differences between processing a 0 and processing a 1. Parameters he can monitor are time, power consumption, noise, …

If one is uncareful then this will give away a for free!

Huge threat, unless a is public anyway (as in signature verification).

Countermeasures:

Adding unnecessary computations, using uniform addition formulas, …
but the problem is somewhat inherent to double-and-add.

Use a Montgomery ladder for scalar multiplication.

Elliptic-curve-specific speed-ups

Using projective coordinates

Remember the addition resp. doubling formula for Weierstrass curve arithmetic:

$$\begin{cases} x_3 = \lambda^2 - x_1 - x_2 \\ y_3 = -y_1 - \lambda(x_3 - x_1) \end{cases} \quad \text{where } \lambda = \frac{y_2 - y_1}{x_2 - x_1} \text{ resp. } \lambda = \frac{3x_1^2 + 2Ax_1}{2y_1}.$$

Each step in the addition/subtraction chain requires a computation of λ, which involves a costly field inversion.

Way around: use projective coordinates, computing

$$P_3 = (x_3 : y_3 : z_3) \text{ from } P_1 = (x_1 : y_1 : z_1) \text{ and } P_2 = (x_2 : y_2 : z_2).$$

Resulting formulas are inversion-free and even less case distinctive!

At the end of the double-and-add iteration, we can do a **single inversion** of the z-coordinate to find a point of the form $(x, y) = (x : y : 1)$, as wanted.

Using projective coordinates

Remember the addition resp. doubling formula for Weierstrass curve arithmetic:

$$\begin{cases} x_3 = \lambda^2 - x_1 - x_2 \\ y_3 = -y_1 - \lambda(x_3 - x_1) \end{cases} \quad \text{where } \lambda = \frac{y_2 - y_1}{x_2 - x_1} \text{ resp. } \lambda = \frac{3x_1^2 + 2Ax_1}{2y_1}.$$

Each step in the addition/subtraction chain requires a computation of λ, which involves a costly field inversion.

Formulas for this are easy to establish: replace $x_1 \leftarrow \frac{x_1}{z_1}, y_1 \leftarrow \frac{y_1}{z_1}, x_2 \leftarrow \frac{x_2}{z_2}, y_2 \leftarrow \frac{y_2}{z_2}$ and put on common denominators. For example in the case of addition this gives:

$$P_3 = ((x_2z_1 - x_1z_2)(y_2z_1 - y_1z_2)^2 z_1z_2 - (x_2z_1 + x_1z_2)(x_2z_1 - x_1z_2)^3 : \ldots : (x_2z_1 - x_1z_2)^3 z_1z_2).$$

Looks ugly, but is more efficient!

Literature contains various clever ways of evaluating these formulas efficiently.
Useful other types of homogeneous coordinates (e.g. weighted).

Other formulas

The formulas for addition and doubling on a Weierstrass curve are not unique. Using the identities

$$y_1^2 = x_1^3 + Ax_1 + B \quad \text{and} \quad y_2^2 = x_2^3 + Ax_2 + B,$$

it is possible to rewrite them.

One possibility: obtain a single formula that works for both addition and doubling! Interesting against side-channel attacks. Example:

$$\begin{cases} x_3 = \dfrac{(x_1x_2 - 2A)x_1x_2 - 4B(x_1 + x_2) + A^2}{(x_1x_2 + A)(x_1 + x_2) + 2y_1y_2 + 2B} \\ y_3 = \dfrac{x_1x_2(x_1 + x_2) - x_3\big((x_1 + x_2)^2 - x_1x_2 + A\big) - y_1y_2 - B}{y_1 + y_2} \end{cases}$$

Remark: new exceptional point pairs will appear, but they are less likely to be hit by an addition/subtraction chain.

Other curve shapes

Weierstrass curves are not the only shapes of elliptic curves that have been studied! Among them other cubics: Hessian curves, Montgomery curves, …

But it's worth even leaving the realm of cubics! Annoying feature for this talk: arithmetic is no longer using tangents and chords.

Most prominent example: if $\mathrm{char}\, k \neq 2$ then we can consider the (twisted) **Edwards curves**

$$ax^2 + y^2 = 1 + dx^2y^2$$

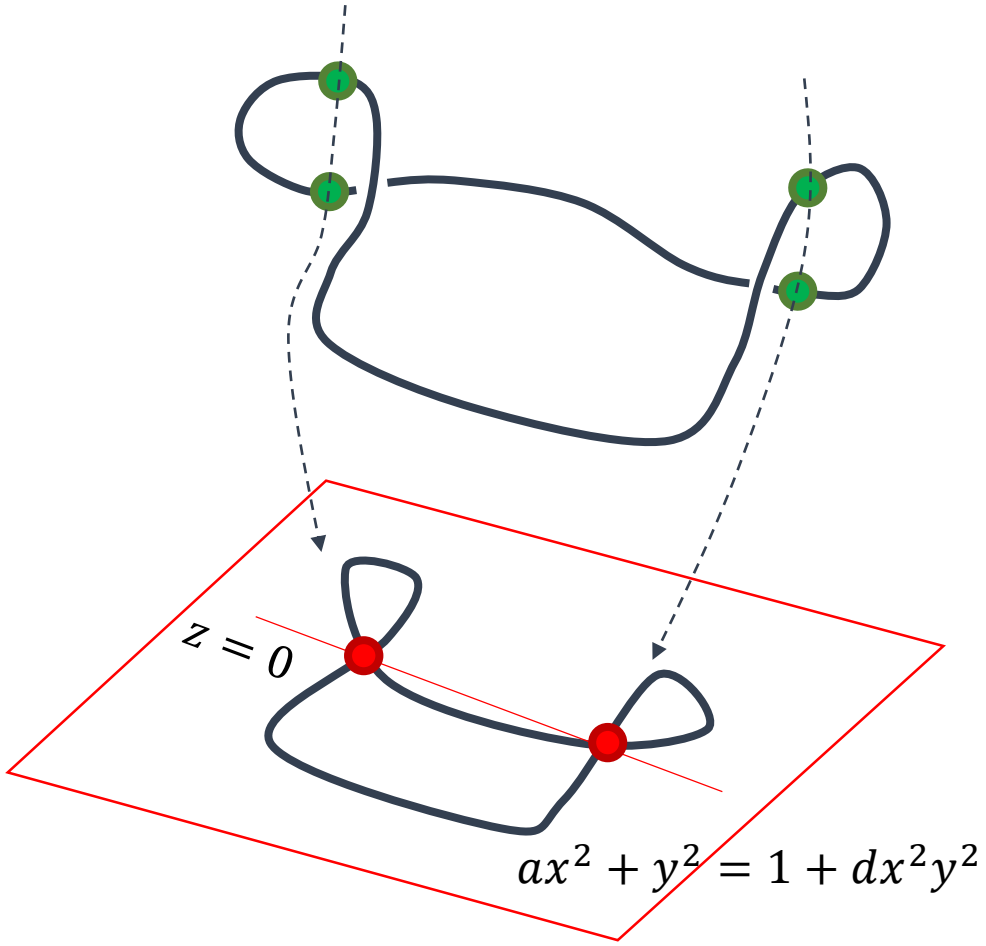
where $a, d \in k$ satisfy $ad(a-d) \neq 0$ and $O = (0,1)$. It admits the amazing addition formula

$$(x_1, y_1) + (x_2, y_2) = \left(\frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2}, \frac{y_1y_2 - ax_1x_2}{1 - dx_1x_2y_1y_2}\right),$$

which are very efficient and can be used for doubling as well (uniformity).

Other curve shapes

A priori annoying aspect of Edwards curves: there are two singular points at infinity, each of which secretly corresponds to **two** points on the complete non-singular model.



But in fact this is a **feature**!

If a is a non-zero square and d is a non-square, then these four points are not defined over k.

Therefore they are never encountered during arithmetic over k, or in other words we have an entirely affine group structure.

Moreover, the addition formula is complete in this case, i.e. it has **no exceptional points**.

x-coordinate only arithmetic and the Montgomery ladder

As we have observed earlier:

$$P \text{ and } Q \text{ have the same } x\text{-coordinate} \quad \Leftrightarrow \quad P = \pm Q$$

Therefore the x-coordinate of aP only depends on the x-coordinate of P, so it should be possible to compute it without any involvement of y-coordinates.

Problem: every double-and-add routine involves addition steps, and there the idea breaks down: $x(P)$ and $x(Q)$ do not suffice to find $x(P+Q)$.

x-coordinate only arithmetic and the Montgomery ladder

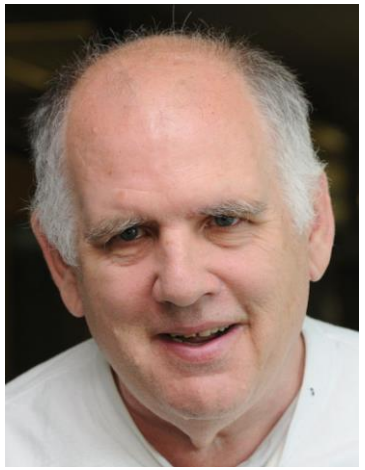
But it *is* true that $x(P+Q)$ is determined by $x(P), x(Q)$ and $x(P-Q)$.

Montgomery found a way to exploit this: recursively compute

$$x(aP), x((a+1)P) \quad \text{from} \quad x\left(\left\lfloor\frac{a}{2}\right\rfloor P\right), x(\left(\left\lfloor\frac{a}{2}\right\rfloor + 1\right)P)$$

using one doubling and one appropriate addition. Note that $x(P)$ is known.

Very fast.
Very uniform: good against side-channel attacks.
Possible to recover the y-coordinate from the end result (Lopez-Dahab).
Comes in projective version: coordinates $(x:z) \in P^1(k)$.
Montgomery chose a more efficient curve form: $By^2 = x^3 + Ax^2 + x$



Peter L. Montgomery

Questions?