The Algebra of Elliptic Curves

Skyler Marks

Boston University

2024-11-12

Intuition

Definition

An **elliptic curve** is the set of pairs of 'numbers' (for an appropriate definition of 'numbers', as we will describe) (x, y) satisfying the equation:

$$y^2 = x^3 + ax + b$$

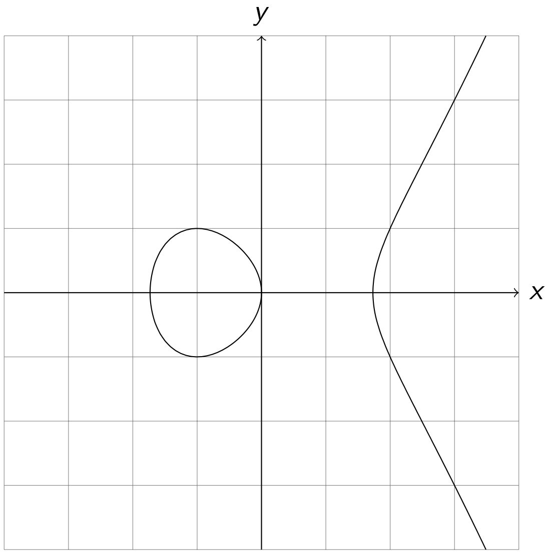


Figure: The graph of the equation $y^2 = x^3 - 3x$.

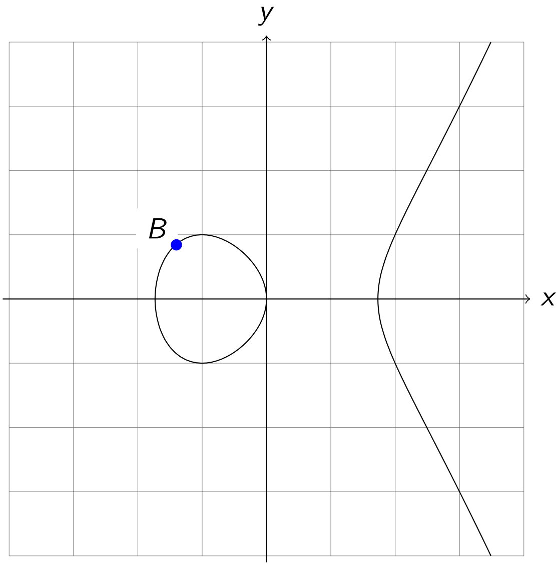


Figure: The first iteration of our process.

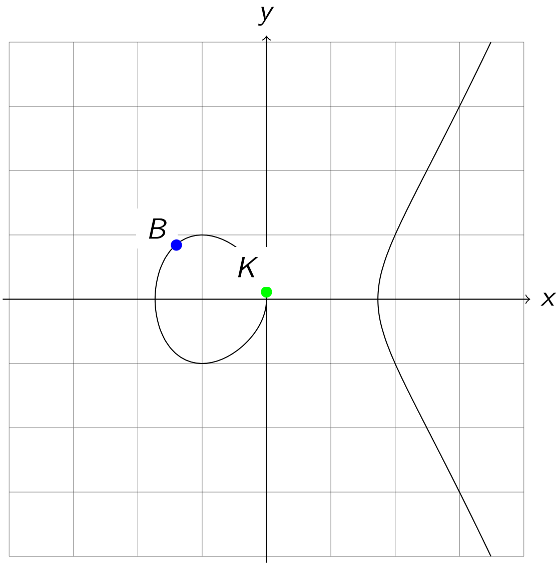


Figure: The first iteration of our process.

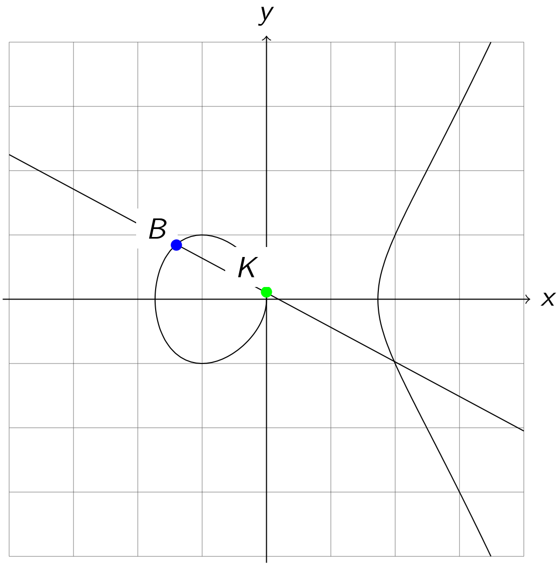


Figure: The first iteration of our process.

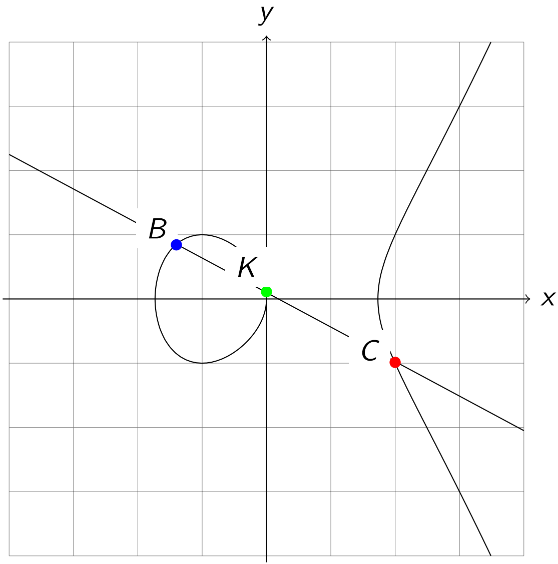


Figure: The first iteration of our process.

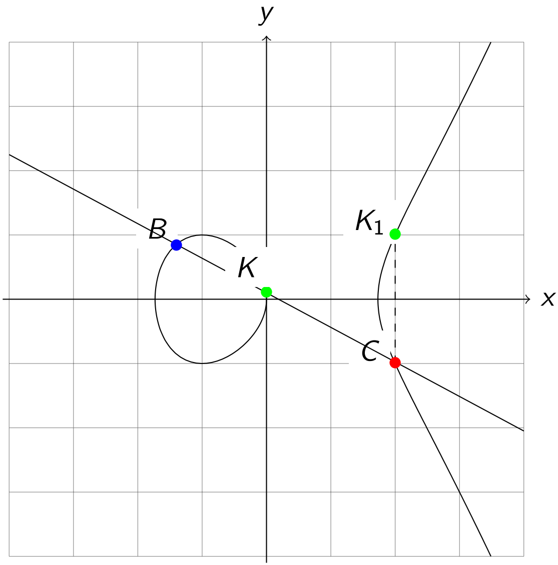


Figure: The first iteration of our process.

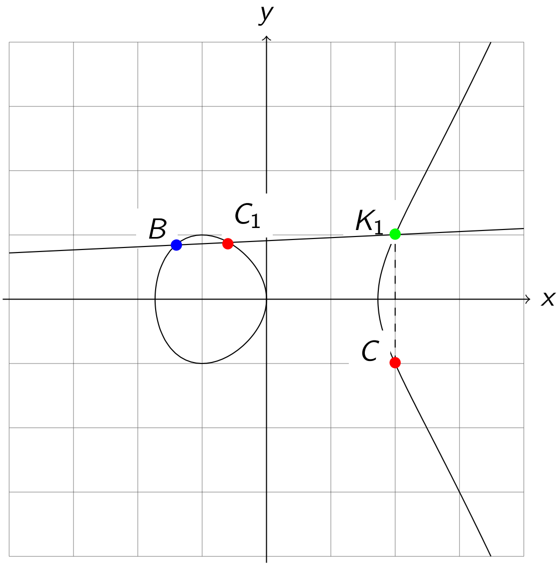


Figure: The second iteration of our process.

A Whirlwind Tour of Abstract Algebra

Definition

For the purposes of this talk, a **set** will be a collection of objects. For more information, consider looking up the Wikipedia page for ZFC (Zermelo - Frankel - Choice set theory, the foundations for most modern math). An element of a set is one of the objects in the set. The notation $a \in A$ means that the object a is in the set A.

Definition

Let A and B be sets. The union of A and B, denoted $A \cup B$, is the set of all elements which are in A or in B (inclusive or). The intersection of A and B, denoted $A \cap B$ is the set of all elements which are in A and in B. The difference $A - B$ or $A \setminus B$ is the set of all elements in A which are not in B.

Definition

The **Cartesian product** of two sets A and B, written $A \times B$, is the set of all pairs (a, b) with $a \in A$ and $b \in B$.

Definition

An **equivalence relation** on a set S is a subset R of $S \times S$ satisfying the following properties:

- Reflexivity: For every a in S, (a, a) is in R.
- Symmetry: If (a, b) in R, then (b, a) is in R.
- Transitivity: If (a, b) is in R and (b, c) is in R, then (b, a) is in R.

We write $a \sim b$ to indicate that (a, b) is in the set R.

Lemma

Let S be a set and $\sim$ be an equivalence relation on S. Let $[a]$ denote the set of all $b \in S$ satisfying $a \sim b$. Every $[a]$ is either equal or disjoint to every other $[b]$, and every element of S is in some $[a]$.

Definition

A **group** is a set G equipped with a binary operation, that is, a function $f : G \times G \to G$. We'll often write the group operation using infix notation using an operator like $\bullet$; $(a \bullet b)$, for example, denotes $f(a, b)$. This binary operation satisfies the following properties:

- Associativity: $a \bullet (b \bullet c) = (a \bullet b) \bullet c$.
- Identity: There is an element e of the set G such that for each g in the set g, $e \bullet g = g \bullet e = g$.
- Inverses: For every g in the set G, there is an element g^{-1} in G satisfying $gg^{-1} = g^{-1}g = e$.

Definition

A group is **abelian** or **commutative** if $a \bullet b = b \bullet a$ for each $a, b \in G$.

Definition

A **subgroup** H of a group G is a subset of G that satisfies the group axioms for the same operation as G.

Definition

A **cyclic subgroup generated by** g for some g in a group G is the set of all 'powers' of g, that is the set of all elements of the form $g \cdot g \cdot \ldots$ or $g^{-1} \cdot g^{-1} \cdot \ldots$, together with the identity.

Example

The symmetries of a triangle are a group.

Example

The integers (under addition) form a group

Example

The integers modulo n form a group under addition.

Example

The integers modulo a prime p, if you take away 0, form a group under multiplication.

Definition

A **ring** is a set R with two binary operations called multiplication and addition, satisfying the following properties:

- Both operations are associative
- The set R is a group under multiplication with identity 0
- There is a multiplicative identity 1
- Multiplication distributes over addition; i.e., for every $a, b, c \in R$

$$a(b + c) = ab + ac \text{ and } (b + c)a = ba + ca$$

A ring is called **commutative** if $ab = ba$ for all a and b in the ring.

Example

The integers are a ring.

Example

The integers mod n are a ring.

Example

Polynomials in n variables with real, integer, or complex coefficients (actually, in any ring) form a ring under the multiplication and addition formulas we're familiar with.

Definition

A **field** is a commutative ring F where the set $F - \{0\}$ is a group under the ring multiplication.

Example

The rational numbers $\mathbb{Q}$, the set of ratios $\frac{p}{q}$ for p, q integers, form a field under the standard 'fraction multiplication'.

Example

The real numbers $\mathbb{R}$ and the complex numbers $\mathbb{Q}$ are fields.

Example

The integers modulo a prime p are a field.

Definition

A **vector space** over a field k is an abelian group V together with an operation $\cdot : k \times V \to V$ called scalar multiplication that is distributive and satisfies $0 \cdot v = \vec{0}$ (where $\vec{0}$ is the identity of the group) and $1 \cdot v = v$.

Example

The Cartesian product $k \times k \times k ... \times k$ is a vector space under componentwise addition and the scalar multiplication law:

$$x \cdot (a_1, a_2, ..., a_n) = (xa_1, xa_2, ..., xa_n)$$

We call this construction **affine n-space over** k

Zero Sets and Projective Space

Definition

Fix a field k. The **zero set** of a polynomial $P(x_1, ..., x_n)$ is the set of all $(x_1, ..., x_n)$ in affine n-space such that $P(x_1, ..., x_n) = 0$.

Definition

Projective n**-space** over a field k is the set of equivalence classes of the set $k \times k \times .... \times k - (0, ..., 0)$ (multiplied $n + 1$ times) by the equivalence relation $a \sim b$ if and only if
$(a_1, ..., a_{n+1}) = \lambda(b_1, ..., b_{n+1})$

Definition

The **degree** of a term in a polynomial is the sum of the powers of the indeterminate variables in that term. A **homogeneous polynomial** in n variables is a polynomial who's terms all have the same degree.

Lemma

The 'zero' of a homogeneous polynomial is a well defined notion in projective space.

Putting it all Together

- Take an **elliptic curve** defined by a polynomial equation $P(x, y)$ over a **finite field** k (for computability).
- Pick a **base point** for our elliptic curve.
- Embed this curve into **projective space** using the homogeneous polynomials associated to $P(x, y)$.
- This yields a group!
- Pick a **private key** (some integer n)
- Now add the base point *to itself* n times - where n is your private key. This yields your **public key**
- Double and Add for speed!

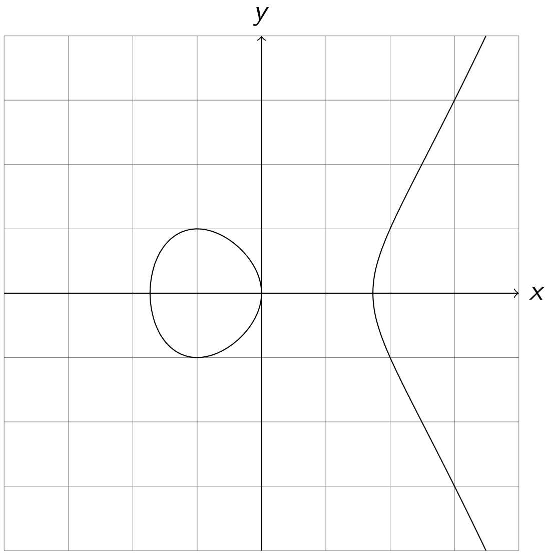


Figure: The graph of the equation $y^2 = x^3 - 3x$.

Computations with TinyEC

Some Python Code for You

```
#!/usr/bin/python

from tinyec import registry
import random

curve = registry.get_curve('secp521r1') # This is the largest prime field key
                                         # recommended by the NSA as of
                                         # recentlyish.
print("There are", curve.field.h, "cyclic groups associated with this field.")
print("""The order of the cyclic group generated on this curve by this
         base point is:,""", curve.field.n)
privKey = random.randint(0, curve.field.n) # random isn't secure but it's fine.
pubKey = privKey * curve.g # This is the beef - generate a public key from a
                           # private key!
print("My public key is:", pubKey)
print("I check that my private key works, and obtain:", privKey * curve.g)
print("I can't read that. Is it equal:", privKey * curve.g == pubKey)
print("My private key is (sshhh, don't tell):", privKey)
```

Let's Time This

```
In [1]: from tinyec import registry

In [2]: import random

In [3]: curve = registry.get_curve('secp521r1')

In [4]: privKey = random.randint(0, curve.field.n)

In [5]: %timeit pubKey = privKey * curve.g

87.1 ms ± 126 s per loop (mean ± std. dev. of 7 runs, 10 loops each)
```

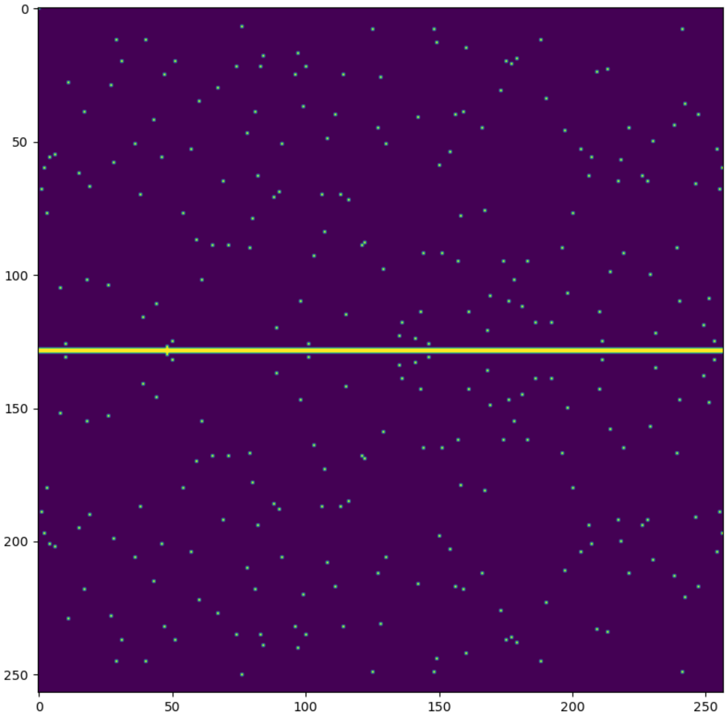
Plotting Elliptic Curves over Finite Fields

Let's plot our equation $y^2 = x^3 - 3x$ over a finite field (in our case, the integers modulo 257).

```
import numpy
import matplotlib.pyplot as plt

p = 257 # To irritate the programmers
grid = numpy.zeros((p, p))
for i in range(0, p):
    for j in range(0, p):
        if (i**2)%p == (j**3-3*j)%p:
            grid[i][j] = 1

for j in range(0, p//2):
    grid[p, j] = 1
plt.imshow(grid, interpolation="nearest")
plt.show()
```


0
50
100
150
200
250