Elliptic-curve groups: a candidate group where the best known discrete log algorithms are the generic ones

↳ Studied by mathematicians since antiquity! [See work of Diophantus, circa 200 AD]

↳ Proposed for use in cryptographic applications in the 1980s ⟶ now is a leading choice for public-key cryptography on the web [another example where abstract concepts in mathematics end up having surprising consequences]

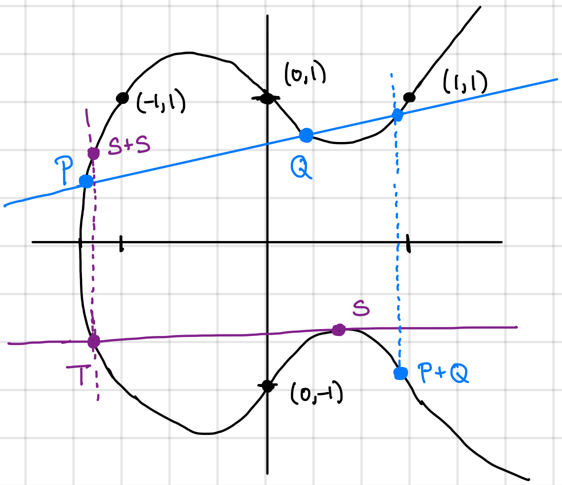
An elliptic curve is defined by an equation of the following form:

$$E: y^2 = x^3 + Ax + B$$

[we will assume that $\underbrace{4A^3 + 27B^2}_{\text{"discriminant" of the curve}} \neq 0$] — non-zero to ensure there are no repeated roots (and the group law is well-defined)

where A, B are constants (over $\mathbb{R}$ or $\mathbb{C}$ or $\mathbb{Q}$ or $\mathbb{Z}_p$)

Example of an elliptic curve: $y^2 = x^3 - x + 1$ (over the reals)



Consider the set of rational points on this curve — points where x- and y-coordinates are rational values

e.g., $(0, \pm 1)$, $(1, \pm 1)$, $(-1, \pm 1)$ [are there other points?]

Surprising facts:

1. Take any two rational points on the curve and consider the line that passes through them. The line will intersect the curve at a new point, which will also have rational coefficients.
2. Take any rational point on the curve and consider the tangent line through that point. The line will intersect the curve at a new point, which will also have rational coefficients.

Thus, given two rational points, there is a way to generate a third rational point.

↳ In fact, this operation essentially defines a group law (but with following modifications):

1. We introduce a "point at infinity" (e.g., a horizontal line at $y = \infty$), denote $\mathcal{O}$ (this is the identity element)
2. The group operation (called the "chord and tangent" method) maps two curve points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ to a point R by first computing the third point that along the line connecting P, Q and reflecting the point about the x-axis. [Observe that the reflection ensures that $\mathcal{O}$ is the identity)

↳ Remarkably, this defines a group law on the rational points on the elliptic curve, and we can write down algebraic relations for computing the group law (somewhat messy but there is a closed form expression)

In cryptography, we work over finite domains, so we instead consider elliptic curves over $\mathbb{Z}_p$ (rather than $\mathbb{R}$ or $\mathbb{C}$).

Specifically, we write

$$E(\mathbb{Z}_p) = \{x, y \in \mathbb{Z}_p : y^2 = x^3 + Ax + B\} \cup \{\mathcal{O}\}$$

No geometric interpretation of the group law over $\mathbb{Z}_p$ (instead, define it using the algebraic definitions derived above)

↳ $E(\mathbb{Z}_p)$ still forms a group under this group law

How big is the group $E(\mathbb{Z}_p)$?

Theorem (Hasse). Let E be an elliptic curve with coefficients in $\mathbb{Z}_p$. Then

$$\big|\, |E(\mathbb{Z}_p)| - (p+1) \,\big| \leq 2\sqrt{p}$$

Thus, number of points on $E(\mathbb{Z}_p)$ is roughly $p \pm \sqrt{p}$

Public-key encryption: Encryption scheme where encryption is public (does not require shared secrets)

- Setup $\to$ (pk, sk) [generates a public/private key-pair — also called KeyGen]
- Encrypt(pk, m) $\to$ c
- Decrypt(sk, c) $\to$ m

(formally, this algorithm takes a security parameter λ, and the public/secret keys are a function of λ)

Everyone can publish a public key (in a directory)

↪ Can encrypt to anyone without exchanging keys (recipient can be offline)

Correctness: $\forall m \in \mathcal{M}: \Pr[(pk, sk) \leftarrow \text{Setup} : \text{Decrypt}(sk, \text{Encrypt}(pk, m)) = m] = 1$

Security: semantic security from secret-key setting, but adversary also gets public key

adversary		challenger
	$b \in \{0,1\}$
	$(pk, sk) \leftarrow$ Setup
$\xleftarrow{\quad pk \quad}$
$\xrightarrow{\quad m_0, m_1 \in \mathcal{M} \quad}$
$\xleftarrow{\quad c_b \leftarrow \text{Encrypt}(pk, m_b) \quad}$
$b' \in \{0,1\}$	

$$\text{SSAdv}[\mathcal{A}, \Pi_{PKE}] = \left|\Pr[\mathcal{A} \text{ outputs } 1 \mid b = 0] - \Pr[\mathcal{A} \text{ outputs } 1 \mid b = 1]\right|$$

In the secret-key setting, we distinguished between semantic security and CPA-security. Here, this is unnecessary since

semantic security $\Rightarrow$ CPA security [means that public-key encryption must be randomized!]

↪ Intuitively: adversary can encrypt messages on its own (using the public key)

PKE from DDH (ElGamal): Let $\mathbb{G}$ be a group with generator g and prime order p

Recall Diffie-Hellman key exchange:

Alice		Bob
$x \xleftarrow{R} \mathbb{Z}_p$	$\xrightarrow{\quad g^x \quad}$	$y \xleftarrow{R} \mathbb{Z}_p$
$\xleftarrow{\quad g^y \quad}$
g^{xy}		g^{xy}

$$\underbrace{g^{xy} \qquad\qquad g^{xy}}_{\text{shared key: } g^{xy}}$$

Idea: Alice will publish $h = g^x$ as her public key

Bob encrypts by choosing fresh share g^y and uses g^{xy} to encrypt the message

security parameter dictates what group is used (e.g., P-256, P-384, P-512)

Setup: $x \xleftarrow{R} \mathbb{Z}_p$, $h \leftarrow g^x$ pk: h, sk: x $\mathcal{M} = \mathbb{G}$, $\mathcal{C} = \mathbb{G}^2$

Encrypt($\overset{=h}{pk}$, m): $y \xleftarrow{R} \mathbb{Z}_p$, $c \leftarrow (g^y, m \cdot h^y)$

Decrypt($\overset{=x}{sk}$, c): $m \leftarrow c_1 / c_0^x$

Correctness:
$$\frac{c_1}{c_0^x} = \frac{m \cdot h^y}{(g^y)^x} = \frac{m \cdot (g^x)^y}{(g^y)^x} = \frac{m \cdot g^{xy}}{g^{xy}} = m$$