The group law in elliptic curves

Elliptic curves. A first not very general definition of elliptic curve over a field K, $\text{char}(K) \neq 2, 3$ is an algebraic curve of the form

$$E:\ y^2 = x^3 + ax + b \qquad \text{with } a, b \in K \quad , \text{ such that } 4a^3 + 27b^2 \neq 0$$

and the 'point at infinity' conventionally denoted by O. It makes sense in the framework of projective geometry.

In Sage there are several forms of introducing an elliptic curve. We consider here the easiest one matching the previous definition: `EllipticCurve(K,[a,b])`

If K is a finite field you can see the points running a `for` loop. For instance

```
E = EllipticCurve(GF(5),[-6,5])
for P in E:
    print P
```

produces the following list of points

```
(0 : 0 : 1)
(0 : 1 : 0)
(1 : 0 : 1)
(2 : 1 : 1)
(2 : 4 : 1)
(3 : 2 : 1)
(3 : 3 : 1)
(4 : 0 : 1)
```

These are the points of $E:\ y^2 = x^3 - 6x + 5$ belonging to $\mathbb{F}_5$, often denoted by $E(\mathbb{F}_5)$. They are in projective notation $(a : b : c)$ means $(a/c, b/c)$ when $c \neq 0$ and the only instance with vanishing last coordinate corresponds to the point at infinity.

Changing the first line to

```
E = EllipticCurve(GF(5^2,'a'),[-6,5])
```

we obtain the result as before plus new points. Remember that $\mathbb{F}_5 \hookrightarrow \mathbb{F}_{5^2}$.

An error is raisen trying to replace $\mathbb{F}_5$ or $\mathbb{F}_{5^2}$ by $\mathbb{F}_7$ or $\mathbb{F}_{7^2}$ because in these fields the condition $4a^3 + 27b^2 \neq 0$ does not hold.

The effect of `show(E)` is displaying the equation of E. This is useless with our presentation of elliptic curves but it is not with others.

The group law. Given P and Q in an elliptic curve over $\mathbb{R}$ we define $P + Q$ as the mirror image respect to the X-axis of the third intersection of the straight line passing through P and Q.

The following code shows it in a picture

```
var('x,y')
graph = implicit_plot(x^3-6*x+5-y^2, (x, -5,5),  (y, -5,5) )
graph += plot( x-1, x, -3,4)
graph += plot( x-1, x, -2,2, thickness=2)
graph += point([1,0],size=30) + point([2,1],size=30)
graph += point([-2,-3],size=30) + point([-2,3],size=30)
graph += line([(-2,3),(-2,-3)], linestyle = '--', thickness=2)
graph += text("P",(2.3,0.4),fontsize=20)
graph += text("Q",(1.3,-0.7),fontsize=20)
graph += text("R",(-2,-3.8),fontsize=20)
graph += text("P+Q",(-2.4,4),fontsize=20)

show(graph)
```

If $P = Q$ we consider that the straight line is the tangent line. If $P = (x, y)$ we define $P = (x, -y)$ and $P + (-P) = O$ (there is not third intersection, it is at infinity). We complete these formulas with $P + O = P$ and $O + P = P$.

It turns out that an elliptic curve E endowed with the operation $+$ is an abelian group.

Using the previous geometric interpretation if $P, Q \neq O$ and $Q \neq -P$ the explicit formula to add $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ is

$$P + Q = (x_3, m(x_1 - x_3) - y_1) \qquad \text{with } m = \frac{y_2 - y_1}{x_2 - x_1} \text{ and } x_3 = m^2 - x_1 - x_2.$$

If $P = Q$ m has to be replaced by $m = (3x_1^2 + a)/2y_1$ which is the slope of the tangent line.

These formulas can be extended to any K (losing the geometrical interpretation) and completed with the trivial cases.

Therefore the following function computes $P + Q$ (the group law)

```
#
# Group law in the elliptic curve y^2= x^3+a*x+b
# ('a' has to be defined in advance)
#
def g_l( P, Q ):
    if P == 'O':
        return Q
    if Q == 'O':
        return P

    if (P[0] == Q[0]) and (P[1] == -Q[1]):
        return 'O'

    if (P[0] == Q[0]) and (P[1] == Q[1]):
        m = (3*P[0]^2+a)/2/P[1]
    else:
        m = (Q[1]-P[1])/(Q[0]-P[0])
    x3 = m^2-P[0]-Q[0]
    return [x3,m*(P[0]-x3)-P[1]]
```

Taking `a=Mod(-6,13)` and `b=Mod(5,13)` (the last one is not necessary) the output of

```
print g_l( [2,1], [2,-1] )
print g_l( [2,1], [1,0] )
print g_l( [2,1], [2,1] )
print g_l( [5,-10], [5,-10] )
print g_l( 'O', [5,-10] )
print g_l(g_l(g_l( g_l( [5,-10],[5,-10] ),[5,-10]),[5,-10]),[5,-10])
```

is

```
O
[-2, 3]
[5, 3]
[2, 12]
[5, -10]
O
```

For instance, the last line means that $P+P+P+P+P=O$. We abbreviate this as $5P=O$.

According to the notation of group theory, we say that P has order 5.

In general the following function gives the order of a point P

```
def ord_p(P):
    k=1
    Q=P
    while Q!='O':
        k += 1
        Q = g_l(P,Q)
    return k
```

but this is not very efficient if the order is large. There are some shortcuts (not discussed here) using for instance the baby-step giant step algorithm.

The command `E.abelian_group()` computes the structure of the abelian group. For instance

```
E = EllipticCurve(GF(5),[-6,5])
print E.abelian_group()
```

inform us that for $E:\ y^2=x^3-6x+5$ over $\mathbb{F}_5$ the group is isomorphic (i.e. the same up to changing names) to $\mathbb{Z}/4\mathbb{Z}\times\mathbb{Z}/2\mathbb{Z}$. The exact output is

```
(Multiplicative Abelian Group isomorphic to C4 x C2, ((3 : 2 : 1), (4 : 0 : 1)))
```

The las part of the output indicates the generators. With

```
a = Mod(-6,5)
b = Mod(5,5)
print ord_p([3,2])
print ord_p([4,0])
```

we check that really the points $(3,2)$ and $(4,0)$ have order 4 and 2, respectively.

Built-in functions in Sage. Actually the functions introduced above in connection to group law are already implemented in Sage.

The point $(x, y) \in E$ in Sage is indicated by `E([x,y])` except the point at infinity that has the special notation `E(0)`.

The sum and the multiplication by an integer is written as usual. For instance the computations that we performed before on the elliptic curve $E: \; y^2 = x^3 - 6x + 5$ over $\mathbb{F}_{13}$ are shortened now with a lighter notation simply as

```
E = EllipticCurve(GF(13),[-6,5])
P = E([2,1])
Q = E([5,-10])

print P+(-P)
print P+E([1,0])
print 2*P
print 2*Q
print E(0)+P
print 5*P
```

Giving the expected result

```
(0 : 1 : 0)
(11 : 3 : 1)
(5 : 3 : 1)
(2 : 12 : 1)
(2 : 1 : 1)
(0 : 1 : 0)
```

The order of P is computed with `P.additive_order()`. Note the new notation with respect to the `multiplicative_order()` that we employed for the group of units of $\mathbb{Z}/N\mathbb{Z}$.

```
E = EllipticCurve(GF(5),[-6,5])
P = E([3,2])
print P.additive_order()
Q = E([4,0])
print Q.additive_order()
```

Gives 4 and 2 as before.