2014-02-03.sagews

February 3, 2014

Contents

1 February 3, 2014: Elliptic Curves, part 1

- whiteboard:
- projector:

1.1 Whiteboard

- Linear equations (one equation)
- Quadratic equations (one equation): Pythagorean triples and that you can enumerate them (how=homework)
- Cubic equations: um, a little bit harder than linear and quadratic focus on elliptic curves for now

1.2 Elliptic Curve Examples

1.2.1 An Elliptic Curve over Q

```
E = EllipticCurve([-5,4])
E
```

```
Elliptic Curve defined by y^2 = x^3 - 5*x + 4 over Rational Field
```

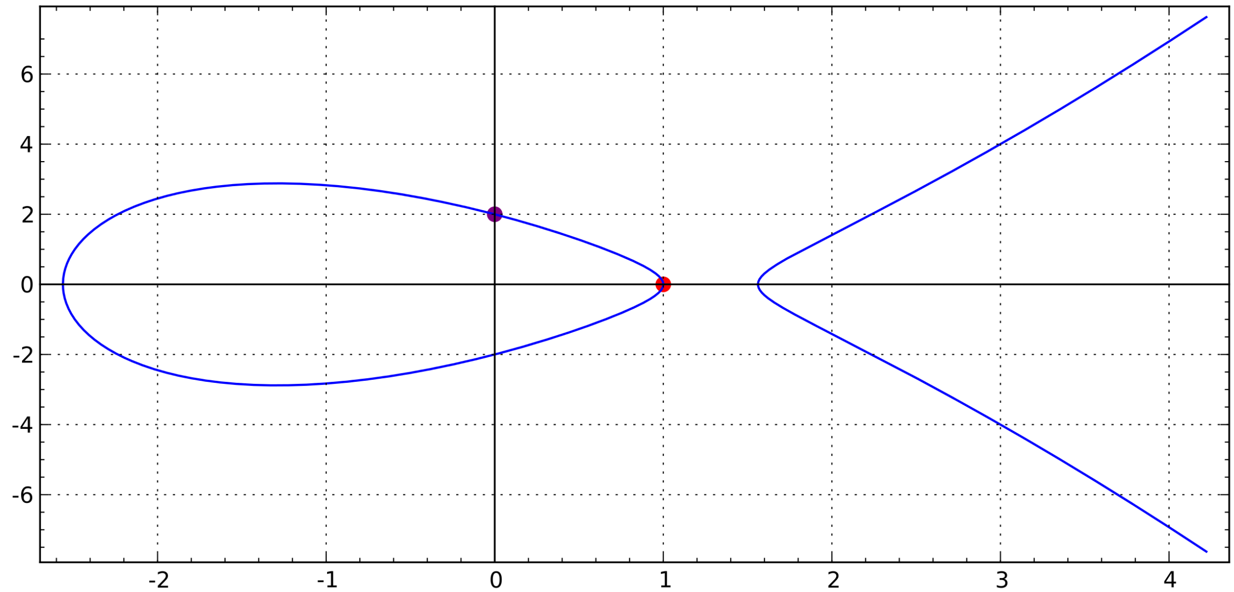
```
# zero element of the group
E(0)
```

```
(0 : 1 : 0)
```

```
# two points
P = E([1,0]); Q = E([0,2])
print "P =", P
print "Q =", Q
```

```
P = (1 : 0 : 1)
Q = (0 : 2 : 1)
```

```
g = plot(E) + point(P[:2],color='red',pointsize=50) + point(Q[:2],color='\
   purple',pointsize=50)
g.show(svg=True, frame=True, gridlines=True)
```




```
P+Q
```

```
(3 : 4 : 1)
```

```
4*Q
```

```
(352225/576 : 209039023/13824 : 1)
```

```
# y^2 = x^3 - 5*x + 4
(209039023/13824)^2 == (352225/576)^3 - 5*(352225/576) + 4
```

```
True
```

```
8*Q
```

```
16*Q
```

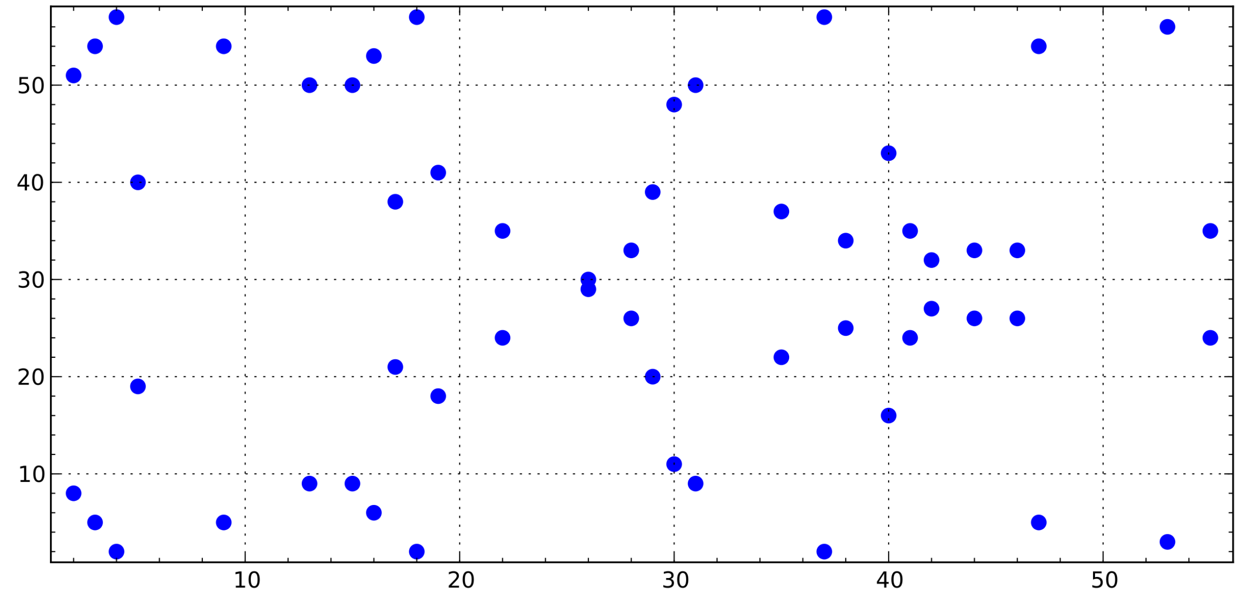
```
32*Q
```

1.2.2 An Elliptic Curve Modulo p

```
E = EllipticCurve(Integers(59), [1,54])
E
```

```
Elliptic Curve defined by y^2 = x^3 + x + 54 over Ring of integers modulo 59
```

```
E.plot(pointsize=50).show(gridlines=True, svg=True, frame=True)
```



```
E.cardinality()
```

```
57
```

```
P = E.points()[5]; Q = E.points()[7]
print "P =", P
print "Q =", Q
```

```
P = (4 : 2 : 1)
Q = (5 : 19 : 1)
```

```
P + Q
```

```
(44 : 26 : 1)
```

1.2.3 Things to come:

- Elliptic curves modulo a huge prime p for creating cryptosystems
- Fake elliptic curves modulo a composite number $n = pm$ for trying to factor
- Elliptic curves over the rational numbers for understanding Diophantine equations such as the one in Fermats Last Theorem

1.3 Computational verification that the group law is associative (over Q, for distinct points).

Our curve is $y^2 = x^3 + ax + b$ and the three points are $(x_1, y_1), (x_2, y_2), (x_3, y_3)$.

```
# Create the polynomial ring in x1-y3 and a,b
R.<x1,y1,x2,y2,x3,y3,a,b> = QQ[]
R
```

```
Multivariate Polynomial Ring in x1, y1, x2, y2, x3, y3, a, b over Rational Field
```

```
# Impose relations
rels = [y1^2 - (x1^3 + a*x1 + b), y2^2 - (x2^3 + a*x2 + b), y3^2 - (x3^3 \
   + a*x3 + b)]
Q = R.quotient(rels)
Q
```

```
Quotient of Multivariate Polynomial Ring in x1, y1, x2, y2, x3, y3, a, b over Rational
Field by the ideal (-x1^3 + y1^2 - x1*a - b, -x2^3 + y2^2 - x2*a - b, -x3^3 + y3^2 - x3*a
- b)
```

```
# Define group operation (assumes points distinct)
def op(P1,P2):
    x1,y1 = P1
    x2,y2 = P2
    lam = (y1-y2)/(x1-x2)
    nu  = y1-lam*x1
    x3  = lam^2 - x1 - x2
    y3  = -lam*x3 - nu
    return (x3, y3)
```

```
# Define points and add them associating both ways
P1 = (x1,y1); P2 = (x2,y2); P3 = (x3,y3)
Z = op(P1, op(P2,P3)); W = op(op(P1,P2),P3)
```

```
# Check that Z and W define the same point
(Q(Z[0].numerator()*W[0].denominator() - Z[0].denominator()*W[0].\
   numerator())) == 0
(Q(Z[1].numerator()*W[1].denominator() - Z[1].denominator()*W[1].\
   numerator())) == 0
```

```
True
True
```