9. Elliptic Curves Cryptography

In the mid-1980s, Miller and Koblitz introduced elliptic curves into cryptograph, and Lenstra showed how to use elliptic curves to factor integers. Since that time, elliptic curves have played an increasingly important role in many cryptographic situations. One of their advantages is that they seem to offer a level of security comparable to classical cryptosystems that use much larger key sizes. For example, it is estimated in that certain conventional systems with a 4096-bit key size can be replaced by 313-bit elliptic curve systems. Using much shorter numbers can represent a considerable saving in hardware implementations.

An elliptic curve E is the graph of an equation

E:$y^2 = x^3 + ax + b$, and denoted by $E_p(a,b)$.

Where a, b are in whatever is the appropriate set(rational numbers, complex numbers, integers mod n, etc.).We also include a **"point at infinity,"** denoted ∞, which is most easily regarded as sitting at the top of the y-axis. It can be treated rigorously in the context of projective geometry, but this intuitive notion suffices for what we need. The bottom of the y-axis is identified with the top , so ∞ also sits at the bottom of the y-axis. When we are working with real numbers, the graph E has one of two possible forms, depending on whether the cubic polynomial in x has one real root or three real roots. In the figure below are shown example of elliptic curves illustrated using Matlab files.

2. $Y^2=x^3-2x+1$

```
x=(-5:0.1:5);
y1=(x).^3-2*x+1;
y2=sqrt(y1);
plot(x,y2);
y3=-y2;
hold on
plot(x,y3);
```

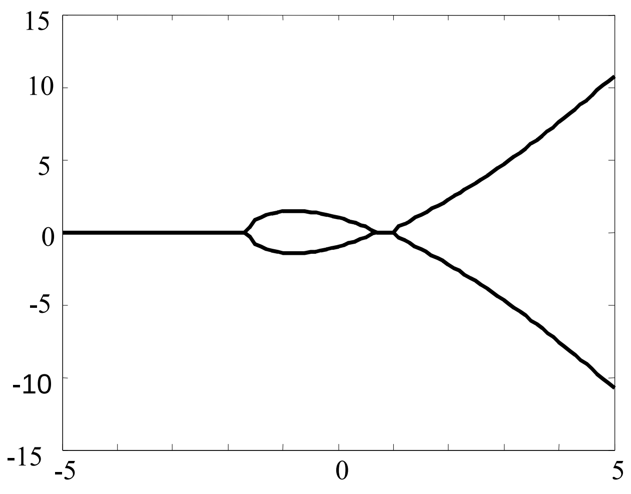


Fig. 9. 1

We assume that the cubic polynomial x^3+ax+b has no multiple roots.

This means we exclude, for example, the graph of $y^2 = x^2(x-1)$.

Technical point :

Given two points P_1 and P_2 on E, we can obtain a third point P_3 on E as follows: Draw the line L through P_1 and P_2 (if $P_1 = P_2$, take the tangent line to E at P_1). The line L intersects E in third point Q. Reflect Q through the x-axis (i.e., change y to -y) to get P_3. Define a law of addition on E by

$$P_1 + P_2 = P_3$$

Note that this is not the same as adding points in the plane.

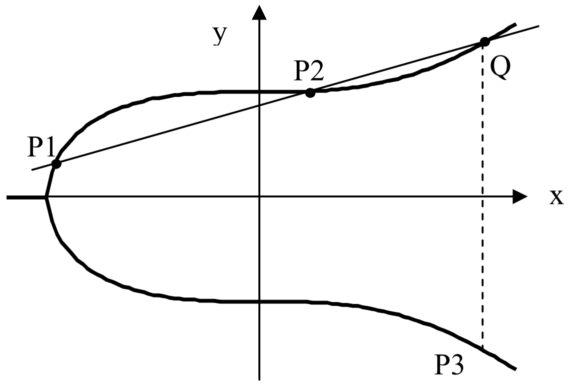


Fig. 8. 2

Example: suppose E is defined by $y^2 = x^3 + 73$. Let $P_1 = (2, 9)$ and $P_2 = (3,10)$. The line L through P_1 and P_2 is $y = x + 7$. Substituting into the equation for E yields $(x + 7)^2 = x^3 + 73$, which yields $x^3 - x^2 - 14x + 24 = 0$. Since L intersects E in P_1 and P_2, we already know two roots, namely $x = 2$ and $x = 3$. Moreover, the sum of the three roots is minus the coefficient of x^2 and therefore equals 1. so the third point of intersection has $x = -4$. since $y = x + 7$, we have $y = 3$, and $Q = (-4, 3)$. Therefore, $P_3 = (-4, -3)$.

Elliptic Curves Mod n

If n is an integer, we can work with elliptic curves mod n using the aforementioned ideas. For example, consider

$$E: y^2 \equiv x^3 + 2x + 3 \pmod 5).$$

The points on E are the pairs (x, y) mod 5 that satisfy the equation, along with the point at infinity. These can be listed as follows. The possibilities for x mod 5 are 0, 1, 2, 3, 4 substitute each of these into the equation and find the values of y that solve the equation.

Historical point: Elliptic curves are not ellipses. They received their name from their relation to elliptic integrals such as

$$\int_{z_1}^{z_2} \frac{dx}{\sqrt{x^3 + ax + b}} \quad \textit{and} \quad \int_{z_1}^{z_2} \frac{xdx}{\sqrt{x^3 + ax + b}}$$

That arise in the computation of the arc length of ellipses.

9.1 Addition operations over elliptic curve

The rules for addition over the elliptic group E p (a,b) are:

1. Let the points P$_1$=(x_1, y_1) and $P_2 = (x_2, y_2)$ be in the elliptic group E_p(a,b), and O is the point at infinity .

$P_1 + P_2 = P_3 = (x_3, y_3)$

$$x_3 = \lambda^2 - x_1 - x_2 \bmod p$$

$$y_3 = \lambda(x_1 - x_3) - y_1 \bmod p$$

$$\text{where slope } \lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & \text{if } P_1 \neq P_2 \\ \dfrac{3x_1^2 + a}{2y_1} & \text{if } P_1 = P_2 \end{cases}$$

2. $P + O = O + P = P$

3. If $x_2 = x_1$ and $y_2 = -y_1$, that is

   P$_1$=(x_1, y_1) and $P_2 = (x_2, y_2) = (x_1, -y_1)$ = - P$_1$, then P$_1$ + P$_2$ = O

Example

E_{23}(1,1), P1=(4,0), P2=(3, 10); P1#P2.

Find $P_1 + P_2 = P_3 = (x_3, y_3)$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} = \frac{10 - 0}{3 - 4} = -10$$

$\lambda = -10$, $x_3 = 100 - 4 - 3 \bmod 23$ =93 mod 23= 1 mod 23;

y_3=-10(4-1)-0 mod 23=-30 mod 23=16mod23

(4,0) + (3, 10) =(1, 16)

9. 2 Multiplication over an elliptic curve group:

The multiplication over an elliptic curve group E_p(a,b) is the equivalent of the modular exponentiation in RSA.

Let P=(3,10)$\in E_{23}$(1,1). Then 2P=(x_3,y_3) is equal to:

$$2P = P + P = (x_1, y_1) + (x_1, y_1)$$

Since $P = Q$ and x_2=x_1, the values of λ, x_3 and y_3 are given by:

$$\lambda = \frac{3x_1^2 + a}{2y_1} \text{ mod p} = \frac{3(3^2)+1}{20} \text{mod } 23 = \frac{5}{20} \text{mod } 23 = 4^{-1} \text{mod } 23 = 6$$

$$x_3 = \lambda^2 - x_1 - x_2 \text{ mod p} = 6^2 - 3 - 3 \text{ mod } 23 = 30 \text{ mod } 23 = 7$$

$$y_3 = \lambda(x_1 - x_3) - y_1 \text{ mod p} = 6(3-7) - 10 \text{ mod } 23 = -34 \text{ mod } 2$$

Therefore $2P = (x_3, y_3) = (7,12)$

In the following table is given the product kP on elliptic curve E23(1,1) for k=1:10.

k	$\lambda = \frac{3x_1 + a}{2y_1}$	x_3 $\lambda^2 - x_1 - x_2 \text{ mod } 23$	y_3 $\lambda(x_1 - x_3) - y_1 \text{ mod } 23$	kP $(x_{3,} y_3)$
1				(3,10)
2	6	7	12	(7,12)
3	12	19	5	(19,5)
4	4	17	3	(17,3)
5	11	9	19	(9,16)
6	1	12	4	(12,4)
7	7	11	3	(11,3)
8	2	13	16	(13,16)
9	19	0	1	(0,1)
10	3	6	4	(6,4)

9. 3 Elliptic Curve ElGamal Encryption

Elliptic curve cryptography can be used to encrypt plaintext messages, M, into ciphertexts. The plaintext message M (Fig. 10.3) is encoded into a point P_M form the finite set of points in the elliptic group, E_p(a,b). The first step consists in choosing a generator point, G$\in E_p(a,b)$, such that the smallest value of n such that n G=O is a very large prime number. The elliptic group Ep(a,b) and the generator point G are made public.

Each user select a private key, n_A<n and compute the public key P_A=$n_A G$. To encrypt the message point P_M for Bob, Alice chooses a random integer k and computes the cipher text pair of points P_C using Bob's public key P_B:

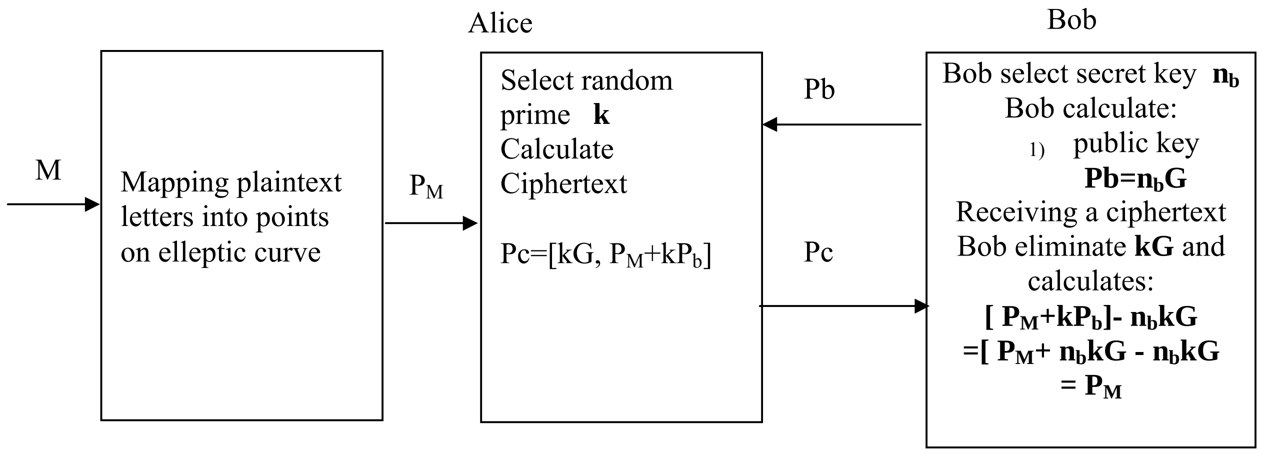
$$P_c=[(KG),(P_M+kP_B)]$$

After receiving the ciphertext pair of points, P_c, Bob multiplies the first point, (kG) with his private key, n_B, and then adds the result to second point in the ciphertext pair of points, $(P_M + kP_B)]$:

$$(P_M + kP_B) - [n_B(kG)] = (P_M + kn_BG) - [n_B(kG)] = P_M$$

which is the plaintext point, corresponding to the plaintext message M. Only Bob, knowing the private key n_B, can remove n_B(kG) from the second point of the ciphertext pair of point, i.e.$(P_M + kP_B)$, and hence retrieve the plaintext P_M.

E_p(a,b) and **G** are public



Alice
Bob
M
Mapping plaintext letters into points on elleptic curve
P_M
Select random prime **k**
Calculate Ciphertext
Pc=[kG, P_M+kP_b]
Pb
Pc
Bob select secret key **n_b**
Bob calculate:
1) public key
Pb=n_bG
Receiving a ciphertext
Bob eliminate **kG** and calculates:
[P_M+kP_b]- n_bkG
=[P_M+ n_bkG - n_bkG
= P_M

Fig. 9. 3

Example

Consider the following elliptic curve:

$$y^2 = x^3 + ax + b \bmod p$$
$$y^2 = x^3 - x + 188 \bmod 751$$

that is : a=-1, b=188, and p=751.The elliptic curve group by the above elliptic curve is then $E_P(a,b) = E_{751}(-1,188)$.

If Alice wants to send to Bob the message M which is encoded as the plaintext point $P_M = (443,253) \in E_{751}(-1,188)$. She must use Bob public key to encrypt it. Suppose that Bob's secret key is $n_B = 85$, then his public key will be:

$$P_B = n_BG = 85(0,376)$$
$$P_B = (671,558)$$

Alice selects a random number k=113 and uses Bob's public key P_B=(671, 558) to encrypt the message point into the ciphertext pair of points:

$$P_C = [(kG),(P_M + kP_B)]$$
$$P_C = [113(0,376),(443,253) + 113(671,558)]$$
$$P_C = [(34,633),(217,606)]$$

Upon receiving the ciphertext pair of points, $P_C = [(34,633),(217,606)]$, Bob uses his private key, $n_B = 85$, to compute the plaintext point, P_M, as follow:

$$(P_M + kP_B) - [n_B(kG)] = (217,606) - [85(34,633)]$$
$$(P_M + kP_B) - [n_B(kG)] = (217,606) - [(47,416)]$$
$$(P_M + kP_B) - [n_B(kG)] = (217,606) + [(47,-416)] \qquad (\sin ce - P = (x_1,-y_1))$$
$$(P_M + kP_B) - [n_B(kG)] = (217,606) + [(47,335)] = \qquad (\sin ce - 416 \equiv 335 (\bmod 751))$$
$$(P_M + kP_B) - [n_B(kG)] = (443,253)$$

and then maps the plaintext point P_M=(443,253) back into the original plaintext message M.

9. 4. Security of ECC

The cryptographic strength of elliptic curve encryption lies in the difficulty for a cryptanalyst to determine the secret random number k from kP and P itself. The fastest method to solve this problem (known as the elliptic curve logarithm problem) is the Pollard ρ factorization method.

The computational complexity for breaking the elliptic curve cryptosystem, using the Pollard ρ method, is $3.8\text{x}10^{10}$ MIPS-years (i.e. millions of instructions per second times the required number of years) or an elliptic curve key size of only 150 bits. For comparison, the fastest method to break RSA, using the General Number Field Sieve Method to factor the composite integer n into the two primes p and q, require $2\text{x}10^{8}$ MIPS-years for a 768-bit RSA key and $3\text{x}10^{11}$ MIPS-years with a RSA key of length 1024.

If the RSA key length is increased to 2048 bits, the General number Field Sieve Method will need $3\text{x}10^{20}$ MIPS-years to factor n whereas increasing the elliptic curve key length to only 234 bits will impose a computational complexity of $1.6\text{x}10^{28}$ MIPS-years(still with the Pollard ρ method).

9. 5. Embedded plaintext to points in Elliptic curve

We can represent letters of the Roman alphabet by distinct points on the elliptic curve, as we have below:

N	1	2	3	4	5	6	7	8	9	10	…
P_M	(3,10)	(7,12)	(19,5)	(17,3)	(9,16)	(12,4)	(11,3)	(13,16)	(0,1)	(6,4)	…
Letter	A	B	C	D	E	F	G	H	I	J	…

In most cryptographic systems, we must have a method for mapping our original message into a numerical value upon which we can perform mathematical operations. In order to use elliptic curves, we need a method for mapping a message onto a point on an elliptic curve. Elliptic curve cryptosystems then use elliptic curve operations on that point to yield a new point that will serve as the ciphertext.

The problem of encoding plaintext messages as points on an elliptic curve is not as simple as it was in the conventional case. In particular, there is no known polynomial time, deterministic algorithm for writing down points on an arbitrary elliptic curve E (mod p).However, there are fast probabilistic methods for finding points, and these can be used for encoding messages. These methods have the property that with small probability they will fail to produce a point. By appropriately choosing parameters, this probability can be made arbitrarily small, say on the order of $1/2^{30}$.Here is one method, due to Koblitz. The idea is the following. Let E:$y^2 \equiv x^3 + ax + b(\bmod p)$ be the elliptic curve. The message m (already represented as a number)will be embedded in the x-coordinate of a point. However, the probability is only about ½ that $m^3 + am + b$ is a square mod p. Therefore, we adjoin a few bits at the end of m and adjust them until we get a number x such that $x^3 + ax + b$ is a square mod p.

More precisely, let K be a large integer so that a failure rate of $1/2^K$ is acceptable when trying to encode a message as a point. Assume that m satisfies (m+1)K<p. The message m will be represented by a number x=mK+j, where $0 \le j < K$. for j=0,1,....,K-1, compute $x^3 ax + b$ and try to calculate the square root $x^3 ax + b(\bmod p)$. For example, if $p \equiv 3(\bmod 4)$, the method of Section 3.9 can be used. If there is a square root y, then we take $P_m = (x, y)$; otherwise, we increment j by one and try again with the new x. We repeat this until either we find a square root or j=K. If j ever equals K, then we fail to map a message to a point. Since $x^3 ax + b$ is a square approximately half of the time, we have about a $1/2^K$ chance of failure.

In order to recover the message from the point $P_m = (x, y)$ we simply calculate m by

m=[x/K],

where [x/K] denotes the greatest integer less than or equal to x/K.

Example. Let p=179 and suppose that our elliptic curve is $y^2 = x^3 + 2x + 7$. If we are satisfied with a failure rate of $1/2^{10}$, then we may take K=10.Since we need mK+K<179, we need $0 \le m \le 16$. Suppose our message is m=5.We consider x of the form mK+j=50+J.The possible choices for x are 50,51,....,59.For x=51 we get $x^3 + 2x + 7 \equiv 121(\bmod 179)$, and

$11^2 \equiv 121 (\text{mod} 179)$. Thus, we represent the message m=5 by the point Pm=(51,11).The message m can be recovered by m=[51/10]=5.

9. 6. Number of Points Mod p

Let E: $y^2 \equiv x^3 + ax + b (\text{mod}\, p)$ be an elliptic curve, where $p \geq 5$ is prime. We can list the points on E by letting x=0, 1,......., p-1 and seeing when x^3+ax+b is a square mod p. Since half of the nonzero numbers are squares mod p, we except that $x^3 + ax + b$ will be a square approximately half the time. When it is a nonzero square, there are two square roots: y and –y .Therefore, approximately half the time we get two values of y and half the time we get no y. Therefore, we expect around p points. Including the point ∞, we except a total of approximately p+1 points. In the 1930s, H. Hasse made this estimates more precise.

Hasse's Theorem. Suppose E (mod p) has N points. Then

$$|N-p-1| < 2\sqrt{p}$$

It can also be shown that whenever N and p satisfy the inequality of the theorem, there is an elliptic curve E mod p with exactly N points.

If p is large, say around 10^{20}, it is infeasible to count the points on an elliptic curve by listing them. More sophisticated algorithms have been developed by Scoof, Atkin, Elkies, and others to deal with this problem.

9. 7. Discrete Logarithms on Elliptic Curves

Recall the classical discrete logarithm problem. We know that $x \equiv g^k$ (mod p) for some k, and we want to find k. There is an elliptic curve version: Suppose we have points A,B on an elliptic curve E and we know that B = kA = (A+A+......+A) for some integer k. We want to find k. This might not look like a logarithm, but it is clearly the analog of the classical discrete logarithm problem. Therefore, it is called the **discrete logarithm problem** for elliptic curves.

There is no good general attack on the discrete logarithm problem for elliptic curves. There is an analog of the Pohling Hellman attack that works in some situations. Let E be an elliptic curve mod a prime p and let n be smallest integer such that nA=∞.If n has only small prime factors, then it is possible to calculate the discrete logarithm k mod the prime powers dividing n and then use Chinese remainder theorem to find k. The Pohling-Hellman attack can be thwarted by choosing E and A so that n has a large prime factor.

9. 8 Factoring with Elliptic Curves

Suppose n=pq is a number we wish to factor. Choose a random elliptic curve mod n and a point on the curve. In practice, one chooses several (around 14 for numbers around 50 digits; more for larger integers) curves with points and runs the algorithm in parallel.

How do we choose the curve ? First , choose a point P and a coefficient a. Then choose b so that P lies on the curve $y^2 = x^3 ax + b.$ This is much more efficient than choosing a and b and then trying to find a point.

For example, let n=2773.Take P=(1,3) and a=4.Since we want $3^2 = 1^3 + 4.1 + b,$ we take b=4.Therefore, our curve is

$$E : y^2 \equiv x^3 + 4x (\bmod 2773).$$

We calculated 2P=(1771,705) in a previous example. Note that during the calculation, we needed to find $6^{-1} (\bmod 2773)$.This required that gcd(6,2773)=1 and used the extended Euclidean algorithm, which was essentially a gcd calculation.

Now let's calculate 3P=2P+P.The line through the points 2P=(1771,705) and P=(1,3) has slope 702/1770.When we try to invert 1770 mod 2773, we find that gcd(1770,2773)=59, so we ca not do this. So what do we do? Our original goal was to factor 2773, so we don't need to do anything more. We have found the factor 59, which yields the factorization 2773=59.47.

Here's what happened. Using the Chinese remainder theorem, we can regard E as a pair of elliptic curves, one mod 59 and the other mod 47.It turns out that $3P = \infty (\bmod 59)$, while $4p = \infty (\bmod 47).$ Therefore, when we tried to compute 3P, we had a slope that was infinite mod 59 but finite mod 47.In other words, we had a denominator that was 0 mod 59 but nonzero mod 47.Taking the gcd allowed us to isolate the factor 59.

The same type of idea is the basis for many factoring algorithms. If n=pq, you cannot separate p an q as long as they behave identically. But if you can find something that makes them behave slightly differently, then they can be found. In the example, the multiplies of P reached ∞ faster mod 59 than mod 47.Since in general the primes p an q should act fairly independently of each other, one would expect that for most curves E (mod pq) and points P, the multiplies of P would reach ∞ mod p and mod q at different times. This will cause the gcd to find either p or q.

Usually , it takes several more steps than 3 or 4 to reach ∞ mod p or mod q. In practice, one multiplies P by a large number with many small prime factors, for example, 10000!..This can be done via successive doubling (the additive analog of successive squaring; see Exercise

10).The hope is that this multiple of P is ∞ either mod p or mod q. This is very much the analog of the p-1 method of factoring. However, recall that the p-1 method (see Section 6.4) usually doesn't work when p-1 has a large prime factor. The same type of problem could occur in the elliptic curve method just outlined when the number m such that mP equals ∞ has a large prime factor. If this happens (so the method fails to produce a factor after a while),we simply change to a new curve E. This curve will be independent of the previous curve and the value of m such that mP=∞ should have essentially no relation to the previous m. After several tries (or if several curves are treated in parallel), a good curve is often found, and the number n=pq is factored. In contrast, if the p-1 method fails, there is nothing that can be changed other than using a different factorization method.

Example. We want to factor n=455839. Choose

$$E: y^2 \equiv x^3 + 5x - 5, P = (1,1)$$

Suppose we try to compute 10!P.There are many ways to do this .One is to compute 2!P,3!P=3(2!P),4!P=4(3!P),.......If we do this, every thing is fine through 7!P, but 8!P requires inverting 599 (mod n).Since gcd(599,n)=599, we can factor n as 599x761.

Let's examine this more closely. A computation shows that E (mod 599) has 640=$2^7 x5$ points and E (mod 761) has 777=3x7x37 points. More over, 640 is the smallest positive m such that mP=∞ on E (mod 599), and 777 is the smallest positive m such that mP=∞ on E (mod 761).Since 8! İs a multiple of 640, it is easy to see that 8!P=∞ on E (mod 599) as we calculated. Since 8!is not a multiple of 777, it follows that 8!P$\neq \infty$ on E (mod 761).Recall that we obtain ∞ when we divide by 0, so calculating 8!P asked us to divide by 0 (mod 599). This is why we found the factor 599.

In general, consider an elliptic curve E (mod p) for some prime p. The smallest positive m such that mP=∞ on this curve divides the number N or a large divisor of N. In any case, if N is a product of small primes, then B! Will be a multiple of N for a reasonably small value of B. Therefore B!P=∞.

A number that has only small prime factors is called **smooth.** More precisely, if all the prime factors of an integer are less than or equal to B, then it is called **B-smooth.** This concept played a role in the quadratic sieve, the p-1 factoring method, and the index calculus attack on discrete logarithms.

Recall from Hasse's theorem that N is an integer near p. It is possible to show that the density of smooth integers is large enough (we'll leave small and large undefined here) that if we choose a random elliptic curve E (mod p), then there is a reasonable chance that the number N is smooth. This means that the elliptic curve factorization method should find p for this choice of the curve. If we try several curves E (mod n), where n=pq, then it is likely that at least one of the curves E (mod p) or E (mod q) will have its number of points being smooth.

In summary, the advantage of the elliptic curve factorization method over the p-1 method is the following. The p-1 method requires that p-1 is smooth. The elliptic curve method requires only that there are enough smooth numbers near p so that at least one of some randomly chosen integers near p is smooth. This means that elliptic curve factorization succeeds much more often than the p-1 method.

The elliptic curve method seems to be best suited for factoring numbers of medium size, say around 40 or 50 digits. These numbers are no longer used for the security of factoring-based systems such as RSA, but it is sometimes useful in other situations to have a fast factorization method for such numbers. For larger numbers, the quadratic sieve and number field sieve are superior