Bitcoin II
&
Introduction to Elliptic Curve Cryptography

Sep. 11, 2019

Overview

- Bitcoin
  - Transactions
- Elliptic Curve Cryptography
  - Introduction
  - Arithmetics
  - Signature

Bitcoin, the protocol

- A blockchain
  - Each block has a definite history
  - Nonce
    - Proof-of-work to make block building hard
  - The Merkle tree stores a set of transactions

Bitcoin, the currency

- A Bitcoin (the currency, ฿) is a number
- Created when mining blocks (finding nonce)
    - The person finding a nonce so that H(block) < ε adds a *Coinbase* transaction
        - Adding 12.5฿ to his wallet (!?!)
            - amount used to larger, will decrease to 6.25 in May 2020
        - 21 million ฿ in total
            - 17.9M in existence
            - Cap will be reached by 2140
        - Afterwards, miners get rewarded in transaction fees only

Bitcoin, the currency

- A Bitcoin (the currency, ฿) is a number
- Created when mining blocks (finding nonce)
  - The person finding a nonce so that H(block) < ε adds a *Coinbase* transaction
  - Adding 12.5฿ to his wallet (!?!)
    - amount used to larger, will decrease to 6.25 in May 2020
  - 21 million ฿ in total
    - 17.9M in existence
    - Cap will be reached by 2140
  - Afterwards, miners get rewarded in transaction fees only

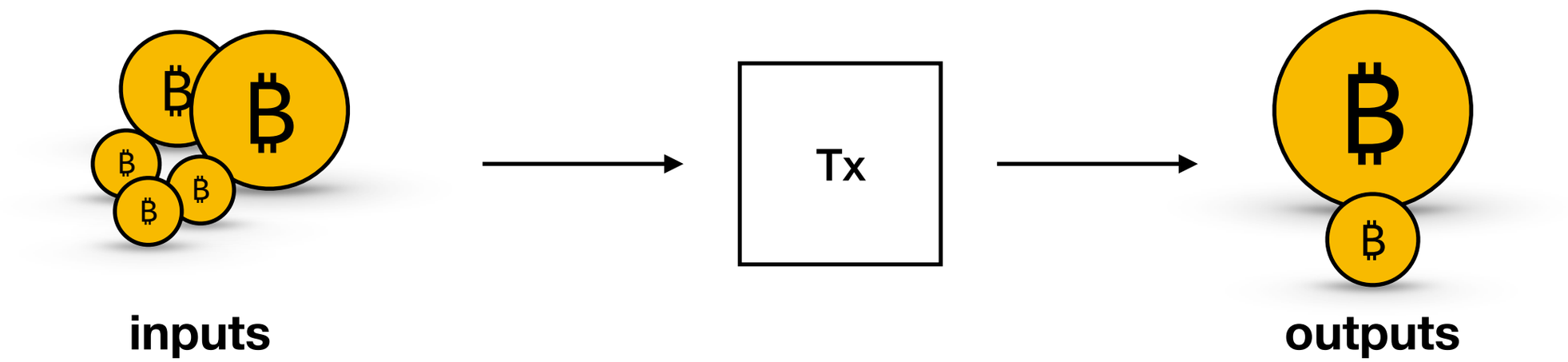
What does this mean?

Bitcoin Ownership

- Ownership is implemented via cryptographic signatures
  - Every person own a public key and a private key
- Ownership means
  - You have the private key of a signature

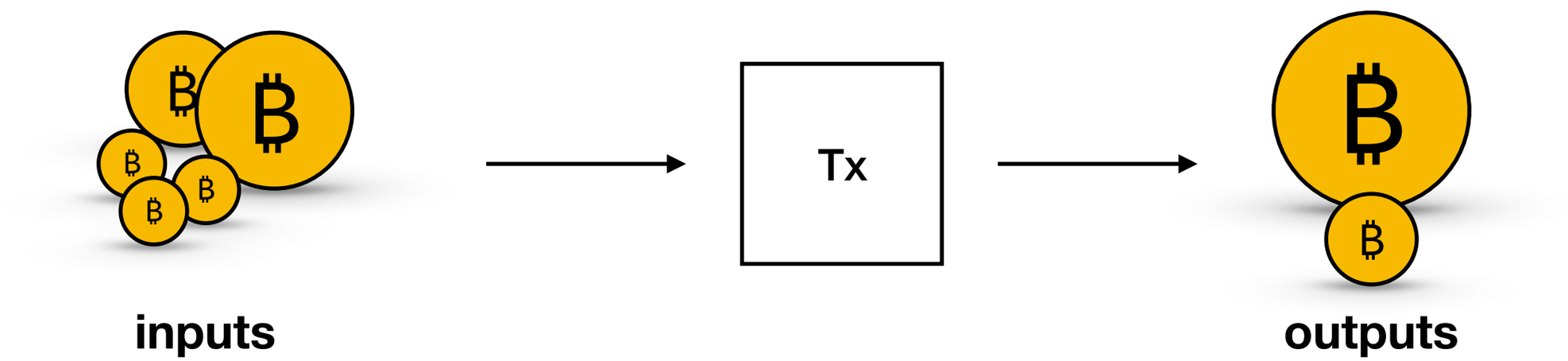
Bitcoin Transactions

- To transfer money to another person, one needs to show
  - Point to (a set of) input amount
  - Demonstrate that you own the input amounts
    - Know private key of input
- Publish a set of outputs

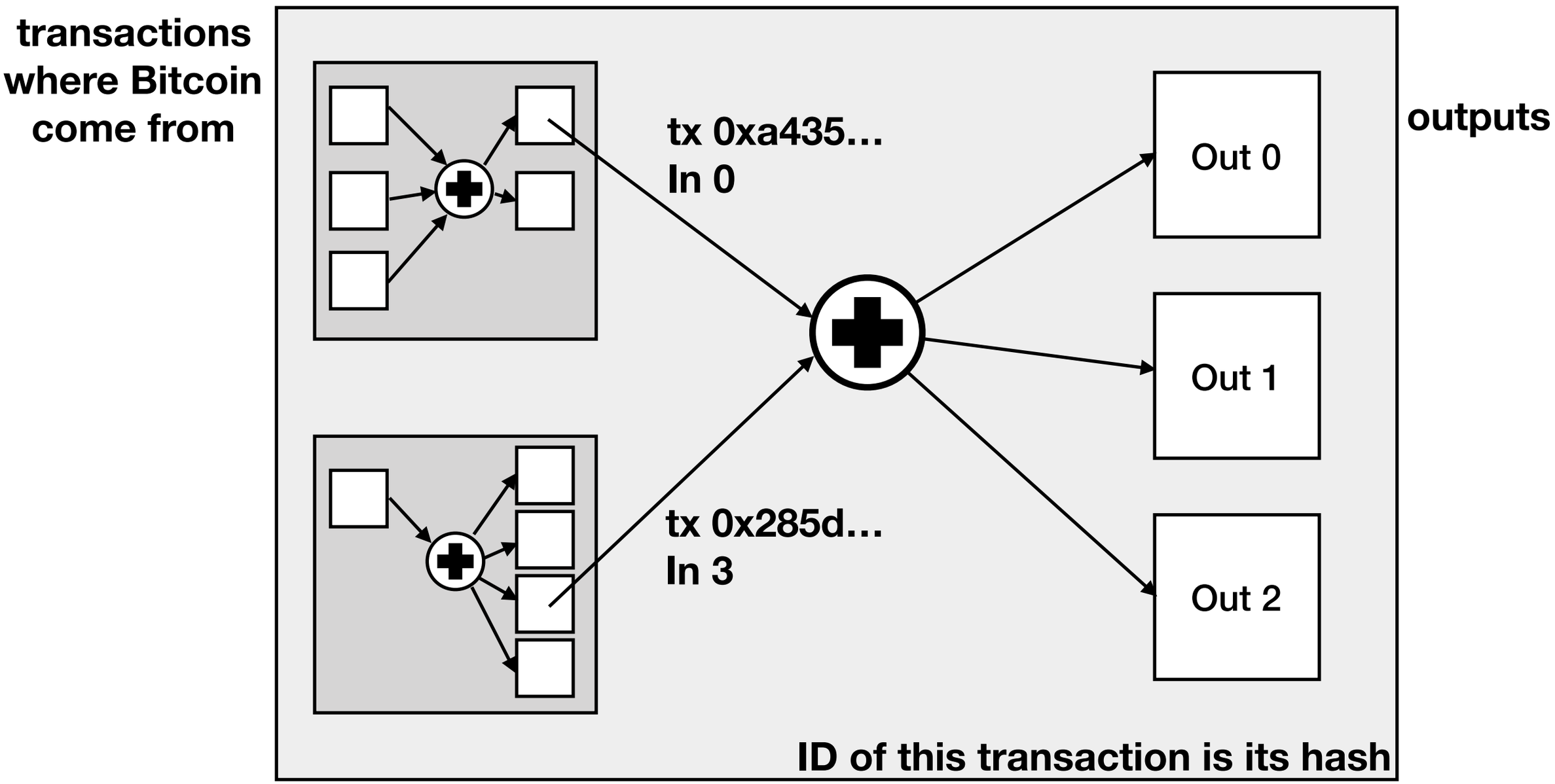


Bitcoin Transactions

- All inputs are completely consumed
- If output is larger than needed, return the rest to you



Bitcoin Transaction



$\sum$ inputs – $\sum$ outputs is fee to miner

Bitcoin Transactions

- Output field:
  - Value
  - Some requirement that has to be fulfilled to claim the output
    - Can be complicated script or open to anyone
- Input fields:
  - Transaction ID
  - Which output in that transaction
  - Proof that the conditions are fulfilled

Transfer money A -> B

- Input In_1 has information about pk_A, public key of A
- A creates a new script, saying that only someone who knows the secret key corresponding to the hash of pk_B (public key of B) can spend the money
    - provide message + public key

Transactions

Input:
Previous tx:
f5d8ee39a430901c91a5917b9f2dc19d6d1a0e9cea205b009ca73dd04470b9a6
Index: 0

scriptSig:
304502206e21798a42fae0e854281abd38bacd1aeed3ee3738d9e1446618c457
90db022100e2ac980643b0b82c0e88ffdfec6b64e3e6ba35e7ba5fdd7d5d6cc8
b241501

Output:
Value: 5000000000
scriptPubKey: OP_DUP OP_HASH160
404371705fa9bd789a2fcd52d2c580b65d35549d
OP_EQUALVERIFY OP_CHECKSIG

Transactions

Input:
Previous tx:
f5d8ee39a430901c91a5917b9f2dc19d6d1a0e9cea205b009ca73dd04470b9a6
Index: 0

scriptSig:
3045022206e21798a42fae0e854281abd38bacd1aeed3ee3738d9e1446618c457
90db022100e2ac980643b0b82c0e88ffdfec6b64e3e6ba35e7ba5fdd7d5d6cc8
b241501

Output:
Value: 5000000000
scriptPubKey: OP_DUP OP_HASH160
4043717 05fa9bd789a2fcd52d2c580b65d35549d
OP_EQUALVERIFY OP_CHECKSIG

Transaction ID showing the funding source

Output 0 within that transaction

Proof of meeting the requirements of that output

Output value (in 1/100,000,000 ฿)

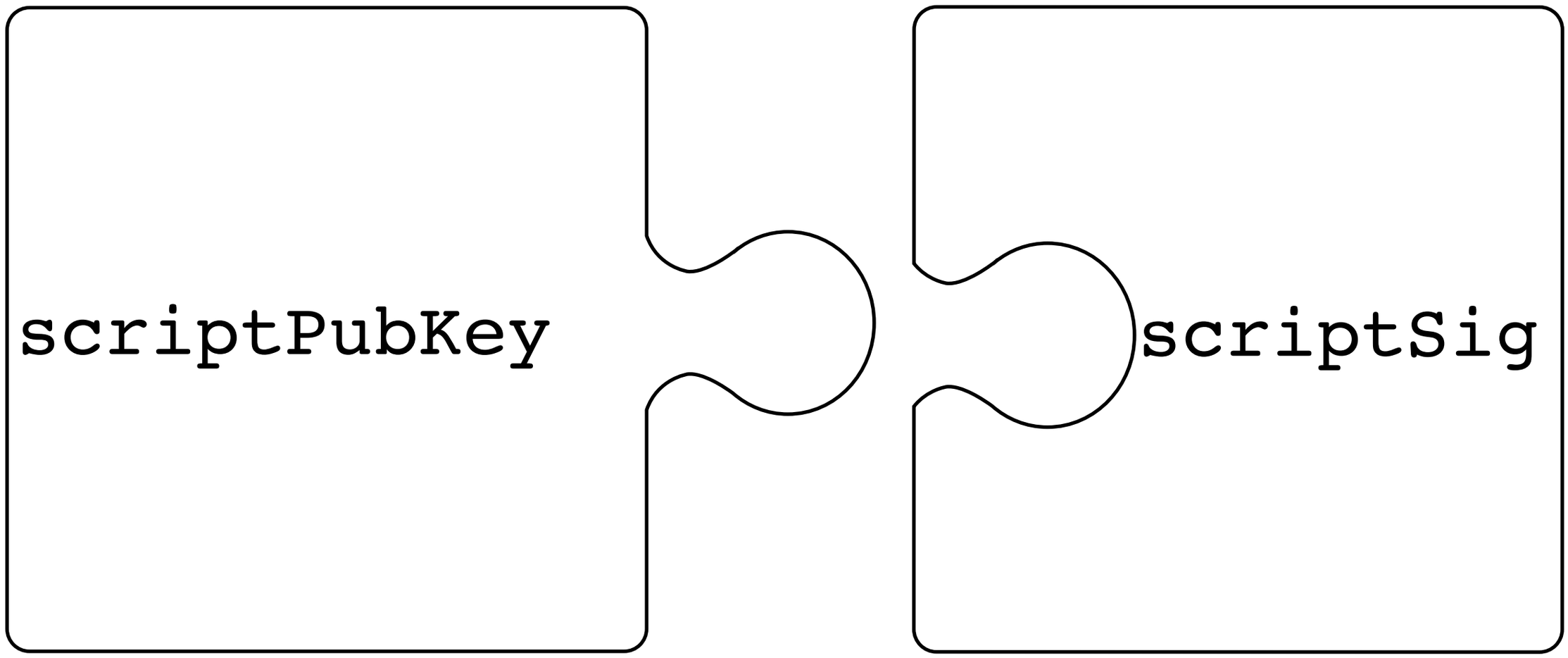
Script to unlock the output value

- This transaction is funded by tx f5d8… output 0, use inputs 3045… and 90db00… as inputs to that script
- Pay 5.00฿ to whoever can run the "OP_DUP…" script successfully.

Transactions

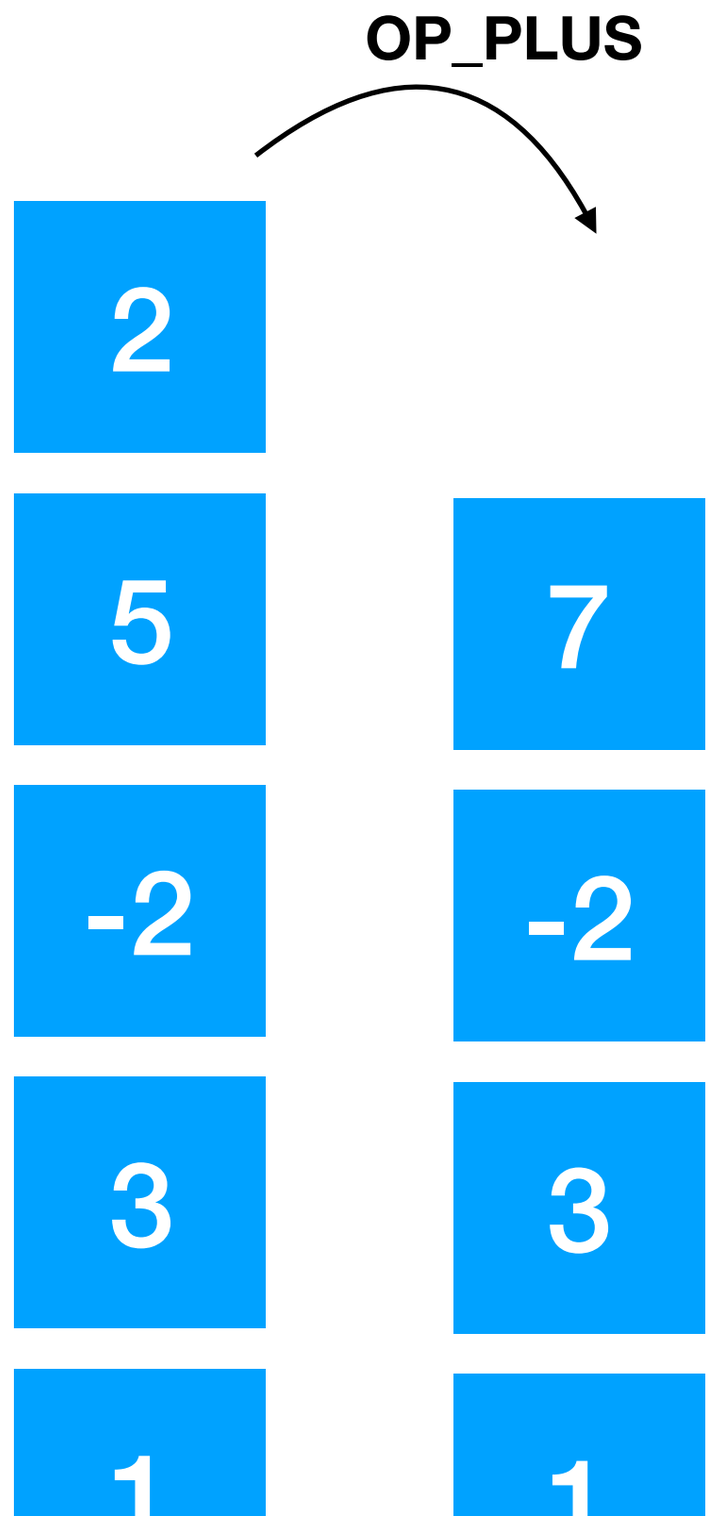
Output of funding transaction specifies script to execute

Input of new transaction provides data for that script



Transaction scripts

- Scripts are executed as a *stack machine*
- Last In - First Out
  - E.g. OP_PLUS : “Take last to element on the stack, add them, and put the result back on the stack”



Transaction scripts
Stack machine

- scriptPubKey:

```
OP_DUP
```
Duplicate element on top of stack

```
OP_HASH160
```
Hash element on top of stack

Push these 20 numbers onto the stack

```
PUSHDATA(20)404371705fa9bd789a2f
```

Verify that the top 2 numbers on the stack are identical

```
OP_EQUALVERIFY
```

Check that the signature is correctly signed by public key

```
OP_CHECKSIG
```

Transaction scripts
Stack machine

- scriptPubKey

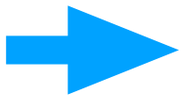
```
OP_DUP
OP_HASH160
PUSHDATA(20)404371705fa9bd789a2f
OP_EQUALVERIFY
OP_CHECKSIG
```

- scriptSig

```
ae0e854281abd38bacd1aeed3ee3e5tadf73
0e88ffdfec6b64e3e6ba35e7ba5fdd7d5d6c
```

Transaction scripts
Stack machine

- scriptPubKey



```
OP_DUP
OP_HASH160
PUSHDATA(20)404371705fa9bd789a2f
OP_EQUALVERIFY
OP_CHECKSIG
```

```
ae0e854281abd38bacd1aeed3ee3e5tadf73
```

```
0e88ffdfec6b64e3e6ba35e7ba5fdd7d5d6c
```

Transaction scripts
Stack machine

- scriptPubKey

```
OP_DUP
OP_HASH160
PUSHDATA(20)404371705fa9bd789a2f
OP_EQUALVERIFY
OP_CHECKSIG
```

```
ae0e854281abd38bacd1aeed3ee3e5tadf73
ae0e854281abd38bacd1aeed3ee3e5tadf73
0e88ffdfec6b64e3e6ba35e7ba5fdd7d5d6c
```

Transaction scripts
Stack machine

- scriptPubKey

```
OP_DUP
OP_HASH160
PUSHDATA(20)404371705fa9bd789a2f
OP_EQUALVERIFY
OP_CHECKSIG
```

```
404371705fa9bd789a2f
ae0e854281abd38bacd1aeed3ee3e5tadf73
0e88ffdfec6b64e3e6ba35e7ba5fdd7d5d6c
```

Transaction scripts
Stack machine

- scriptPubKey

```
OP_DUP
OP_HASH160
PUSHDATA(20)404371705fa9bd789a2f
OP_EQUALVERIFY
OP_CHECKSIG
```

```
404371705fa9bd789a2f
404371705fa9bd789a2f
ae0e854281abd38bacd1aeed3ee3e5tadf73
0e88ffdfec6b64e3e6ba35e7ba5fdd7d5d6c
```

Transaction scripts
Stack machine

- scriptPubKey

```
OP_DUP
OP_HASH160
PUSHDATA(20)404371705fa9bd789a2f
OP_EQUALVERIFY
OP_CHECKSIG
```

```
ae0e854281abd38bacd1aeed3ee3e5tadf73
0e88ffdfec6b64e3e6ba35e7ba5fdd7d5d6c
```

Transaction scripts
Stack machine

- `scriptPubKey`

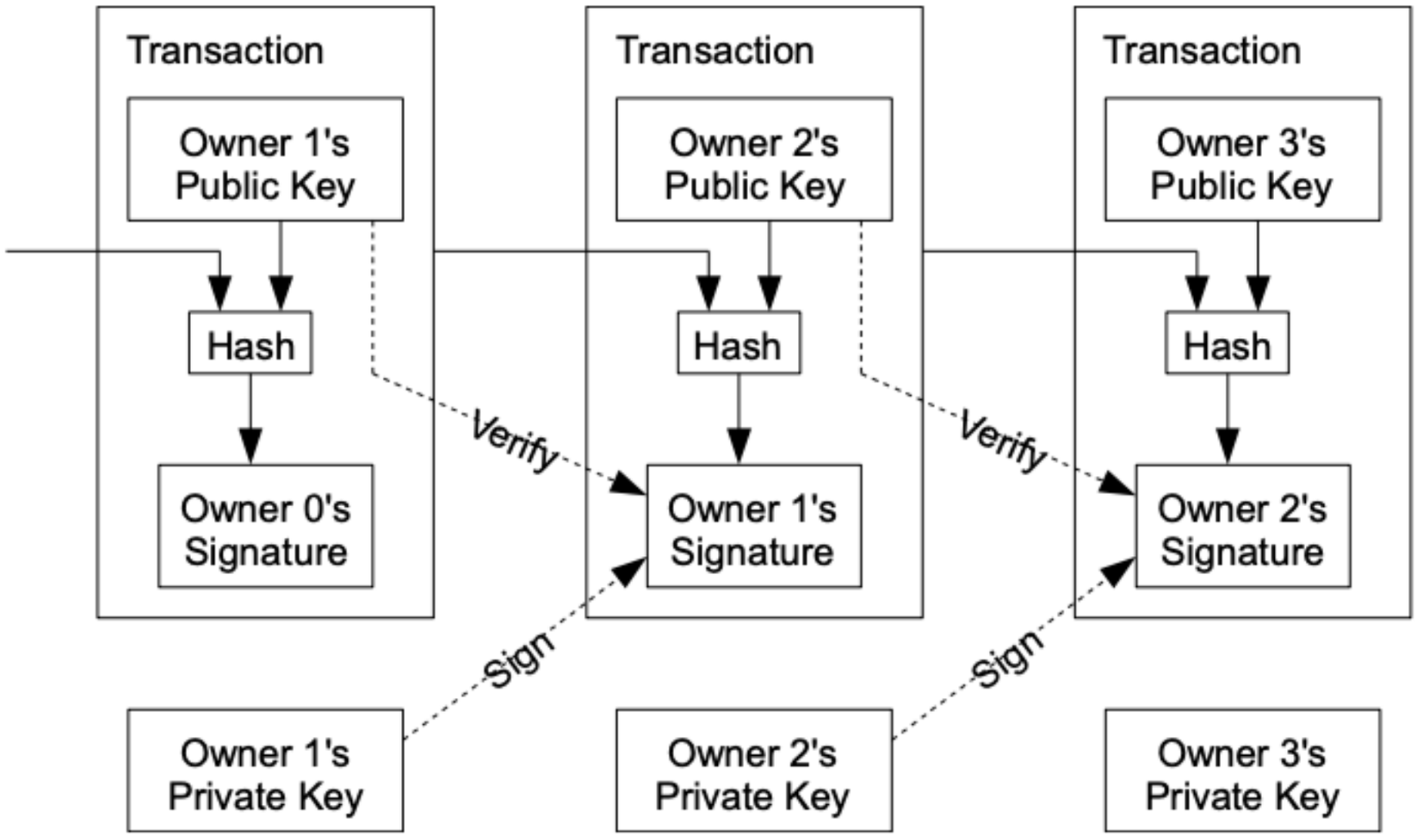
```
OP_DUP
OP_HASH160
PUSHDATA(20)404371705fa9bd789a2f
OP_EQUALVERIFY
OP_CHECKSIG
```

OK if signature

- public key = `ae0e...`
- message = Hash(this transaction)
- signature = `0e88ff...`

is correct

Transaction Chain



Transactions

- Pay-to-Pubkey-Hash
  - Provide a public key and a signature to claim money
- Pay-to-Script-Hash
  - Provide input to arbitrary script
  - preimage (given y, pay to whoever knows x with y=H(x))
  - Long list of other operators

Code	Description
OP_1ADD	1 is added to the input.
OP_1SUB	1 is subtracted from the input.
OP_2MUL	The input is multiplied by 2.disabled.
OP_HASH256	The input is hashed two times with SHA-256.
OP_CHECKSIG	The signature must be a valid signature hash(tx) and public key.
OP_CHECKMULTISIG	Compares the first signature against each public key

Elliptic Curve Cryptography

A basic introduction

What is an elliptic curve

- Consider the polynom

$$x^3 + ax + b$$

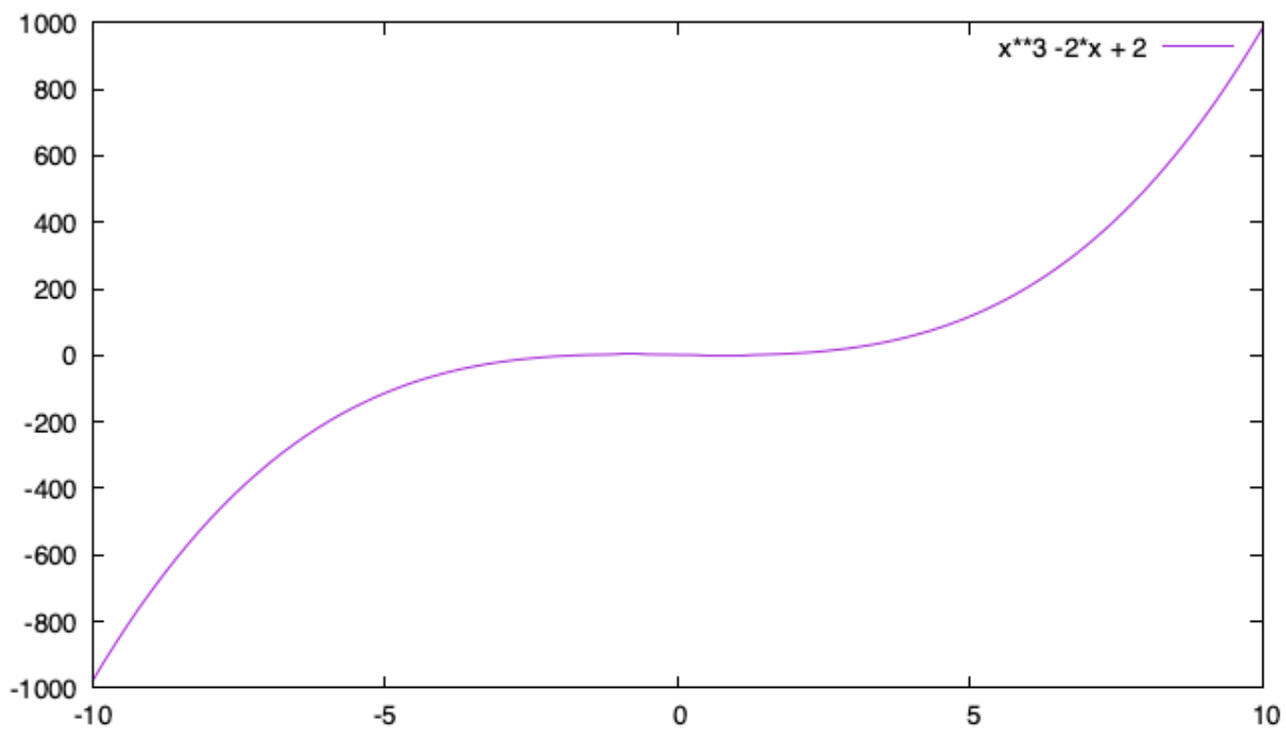
What is an elliptic curve

- Consider the polynom

$$x^3 + ax + b$$

  - For $a = -2$ and $b = 2$ we get

$$y = x^3 - 2a + 2$$



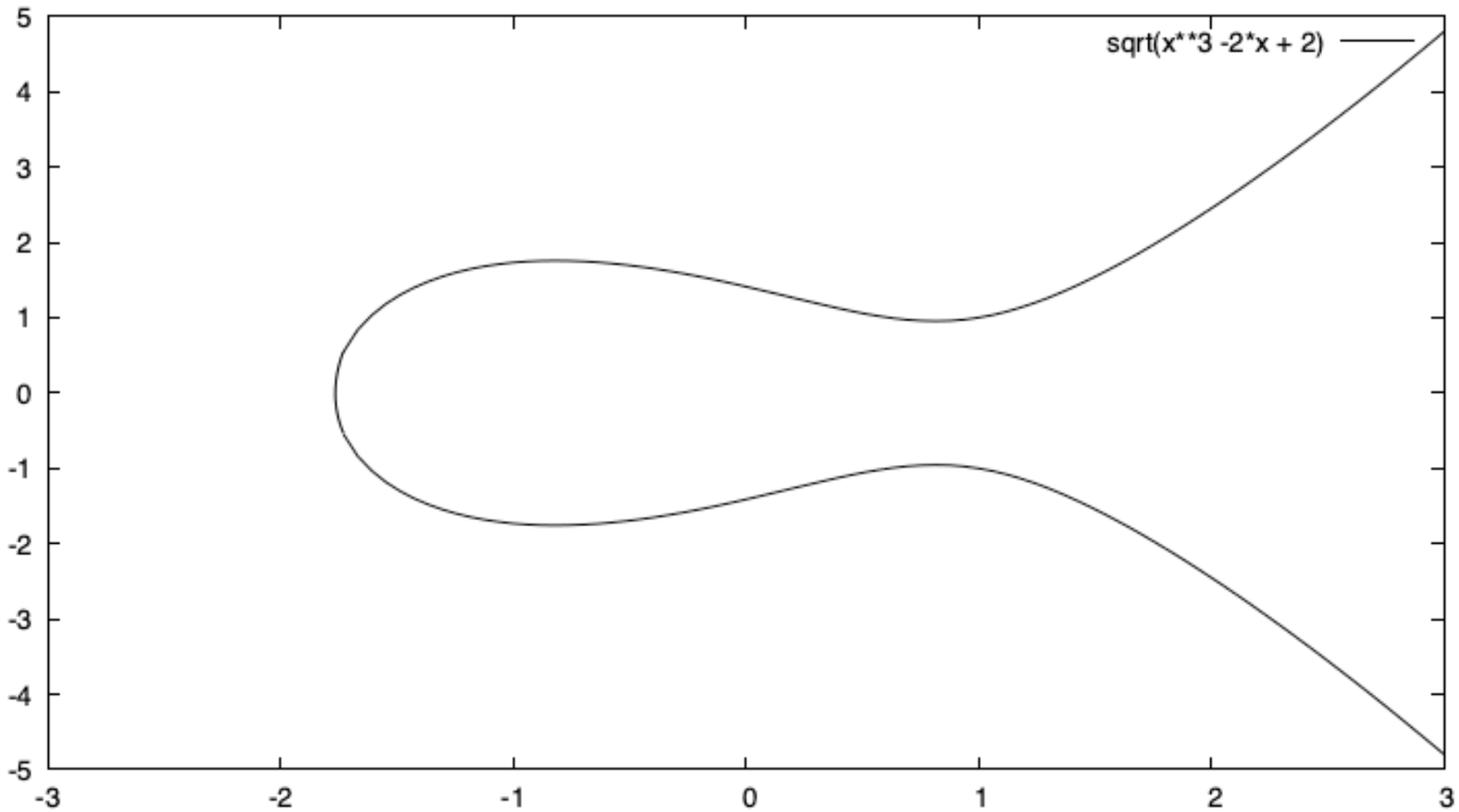
What is an elliptic curve

- For which points on the xy-plane do we have

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

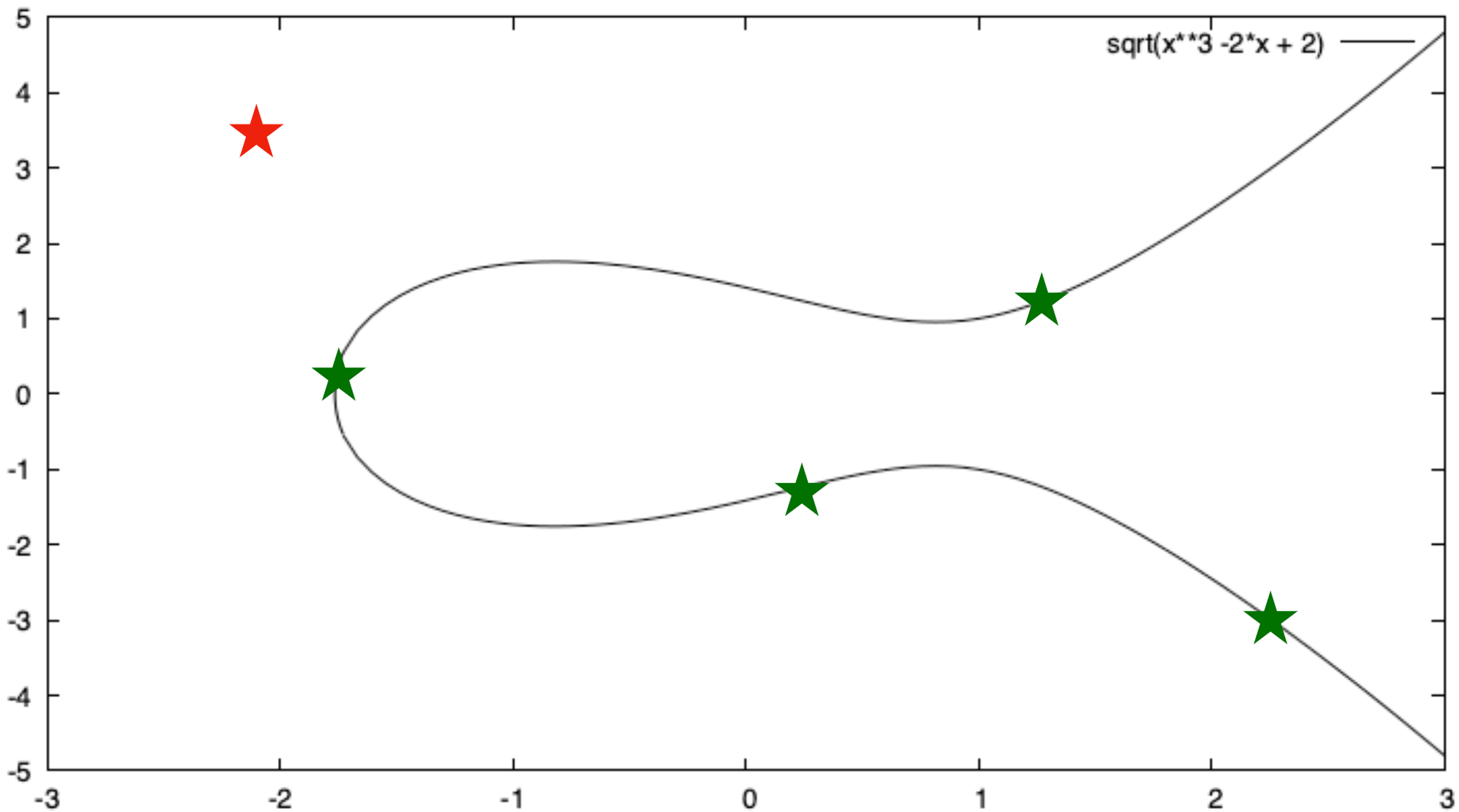
  - For $a = -2$ and $b = 2$ we get

$$y^2 = x^3 - 2a + 2$$



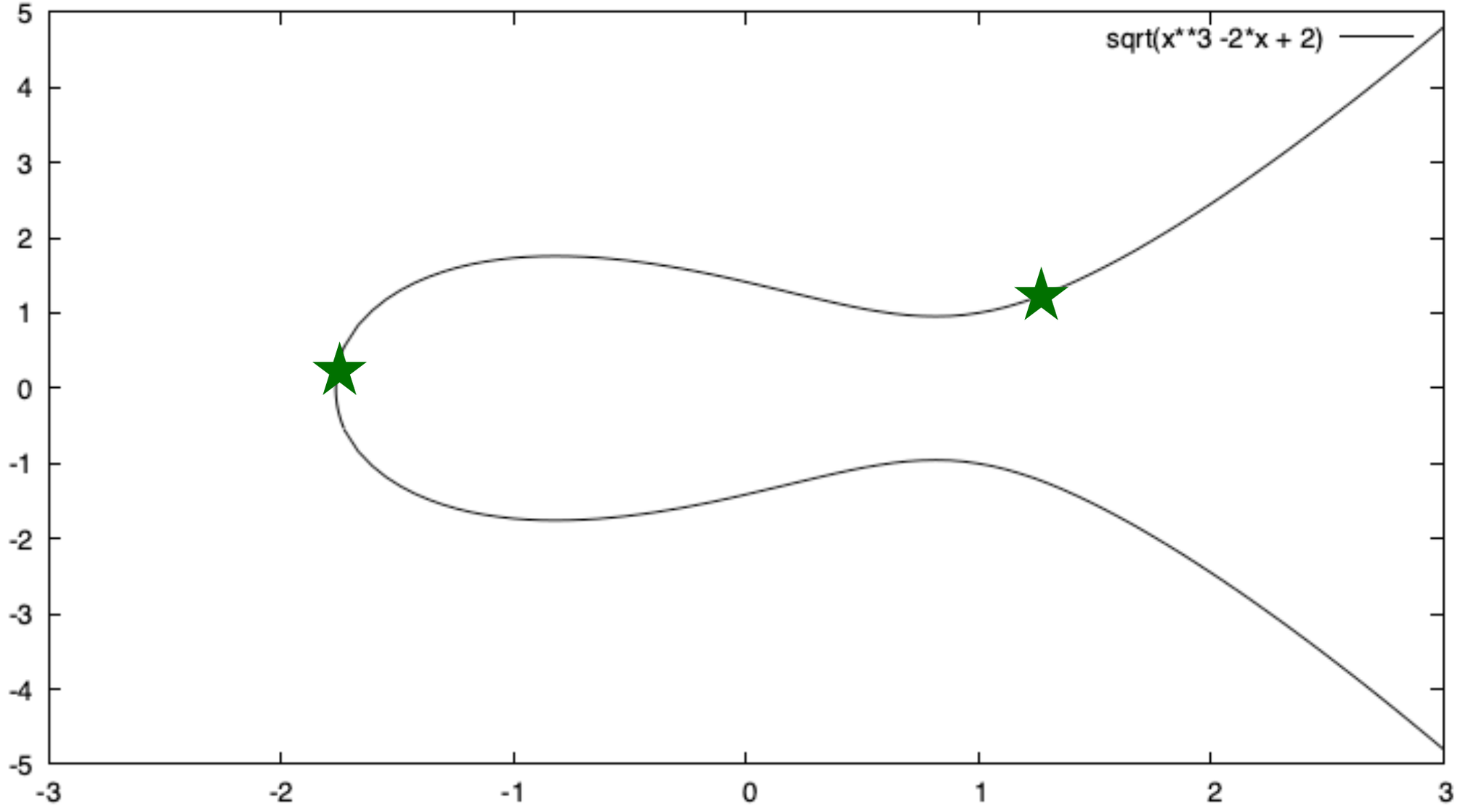
Points on an elliptic curve

- Any point $p = (x, y)$ for which $y^2 = x^3 + ax + b$ is a point
  - Easy to verify



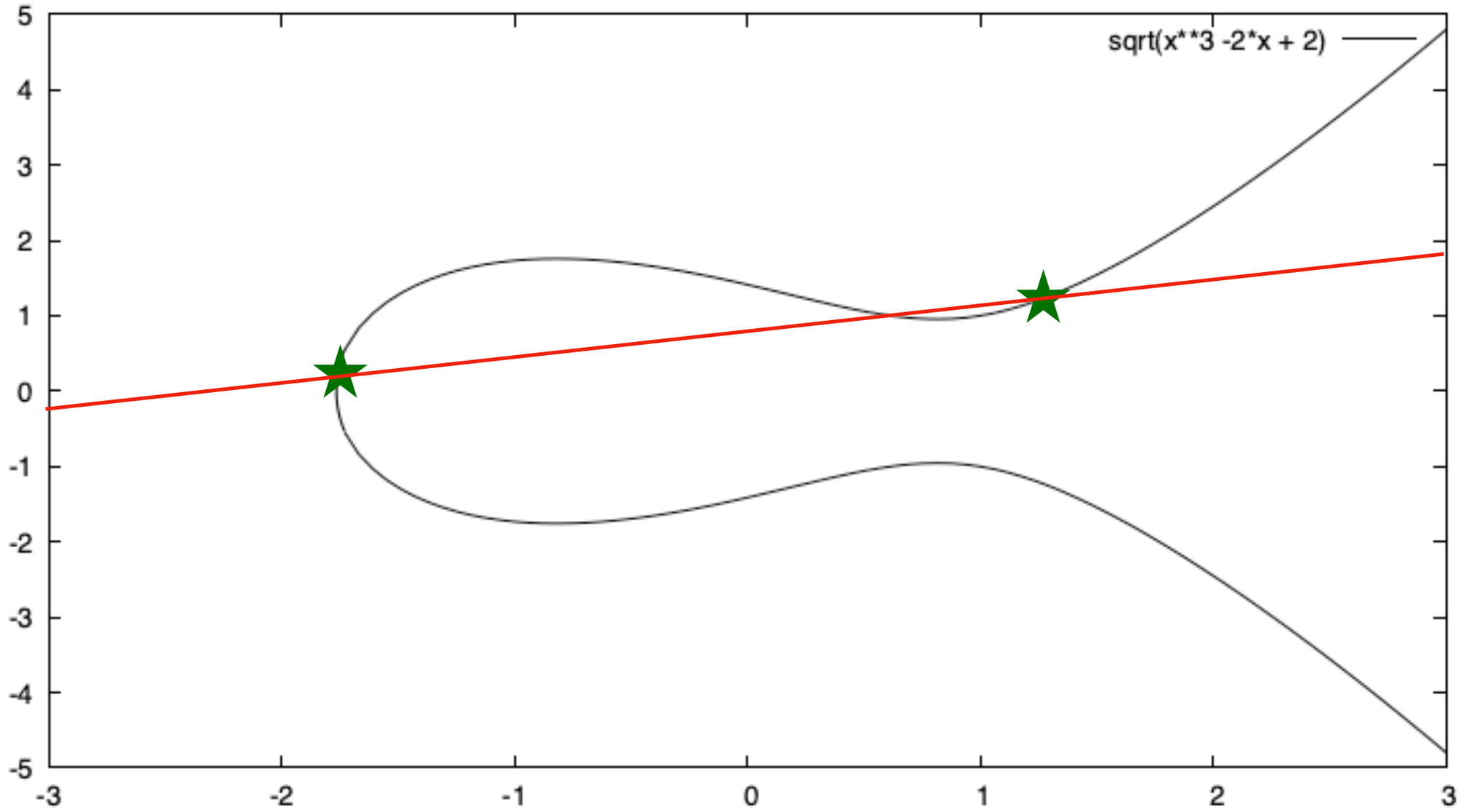
Adding points

- Given 2 points $p = (x, y)$ and $q = (u, v)$



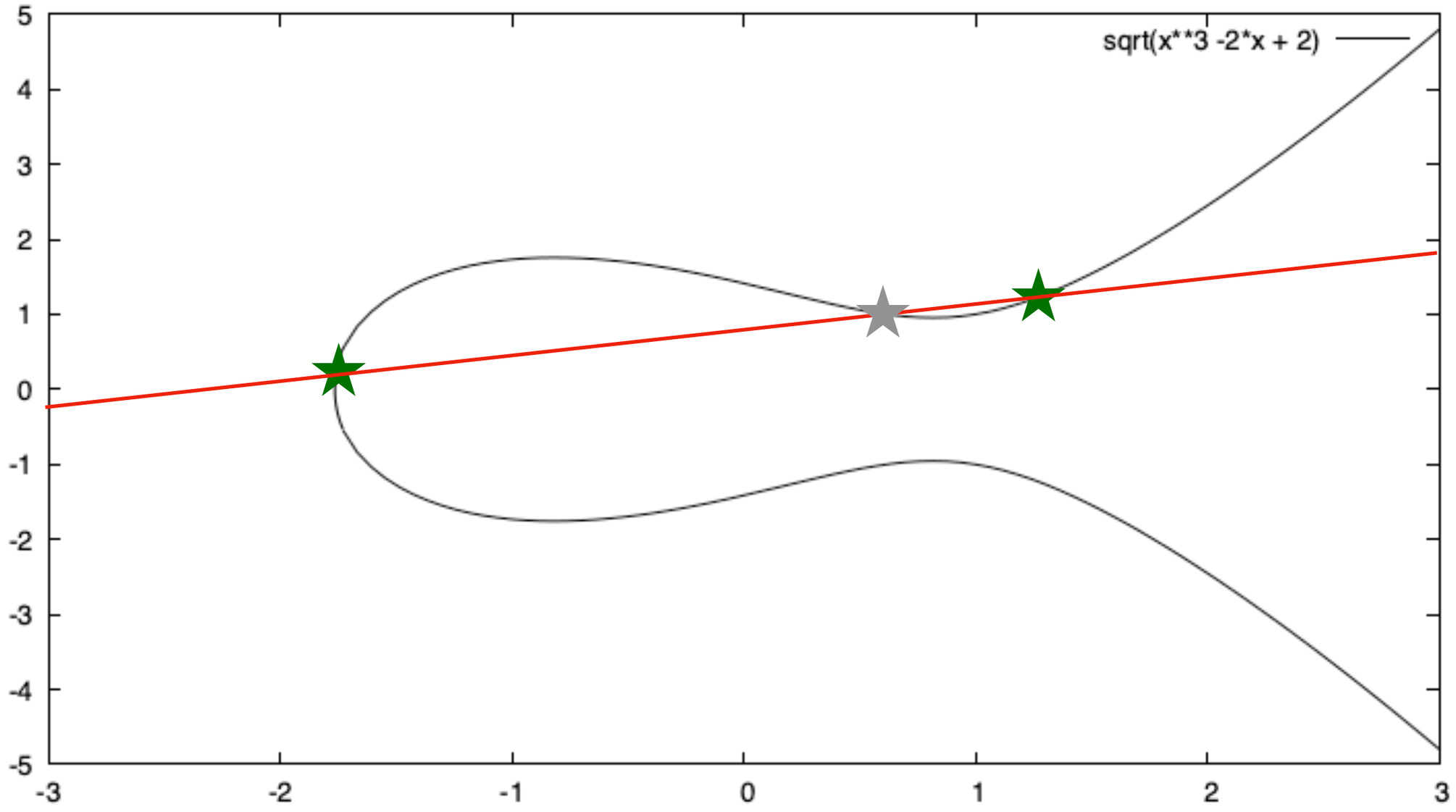
Adding points

- Given 2 points $p = (x, y)$ and $q = (u, v)$
  - Draw a line crossing those points



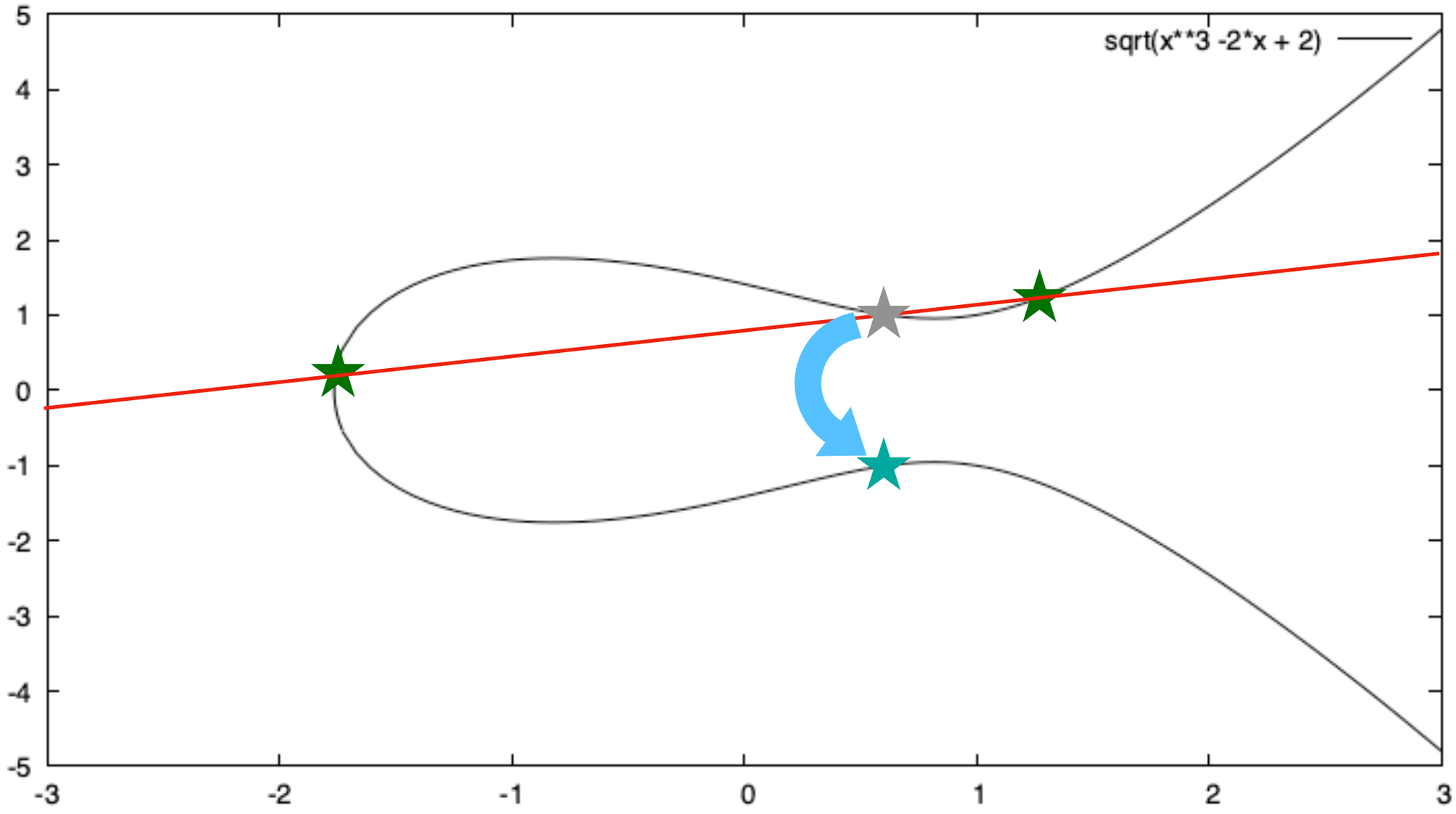
Adding points

- Given 2 points $p = (x, y)$ and $q = (u, v)$
  - Draw a line crossing those points
  - Mark the 3 point of intersection with the curve



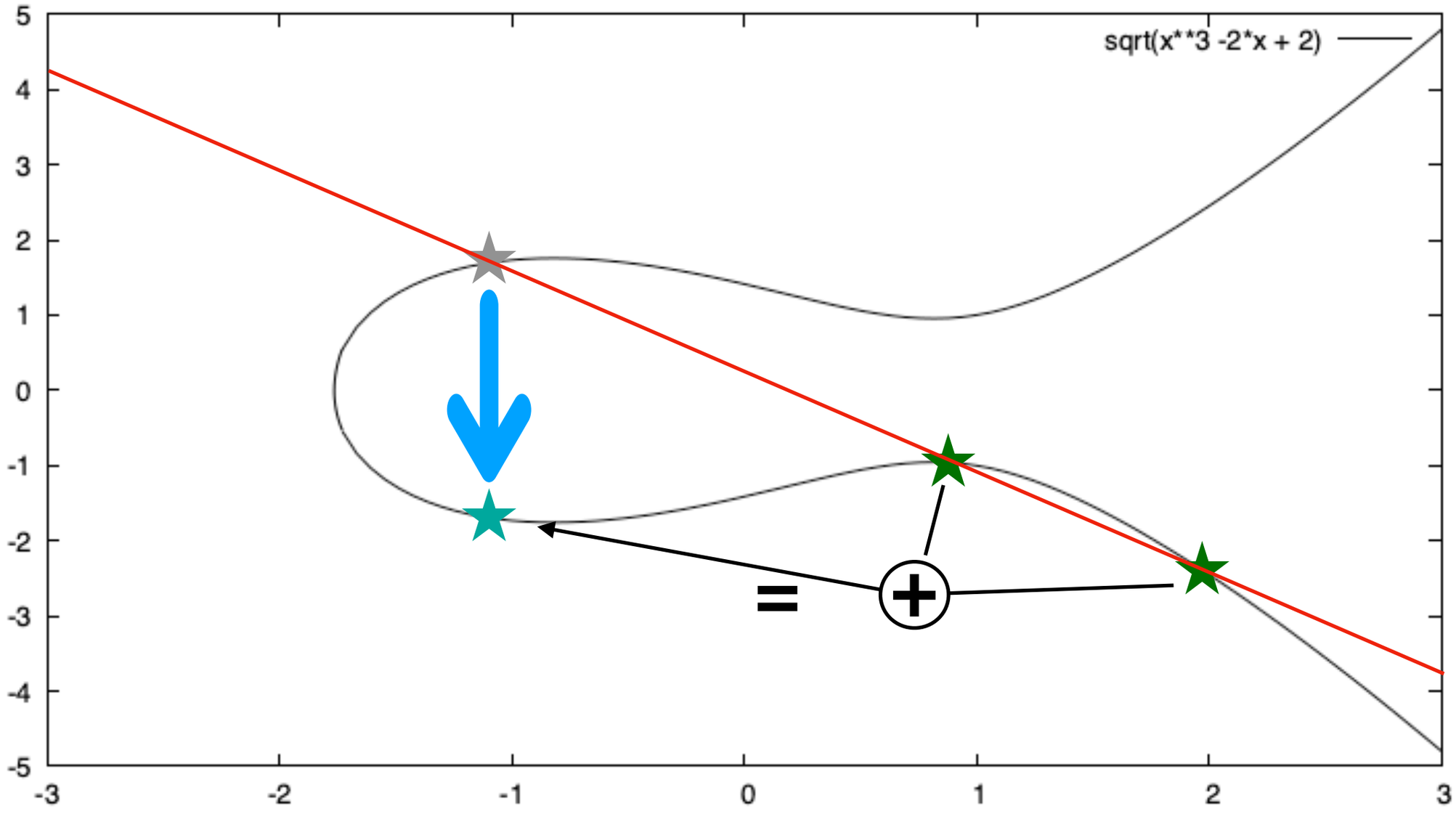
Adding points

- Given 2 points $p = (x, y)$ and $q = (u, v)$
  - Draw a line crossing those points
  - Mark the 3 point of intersection with the curve
  - Flip the point up/down



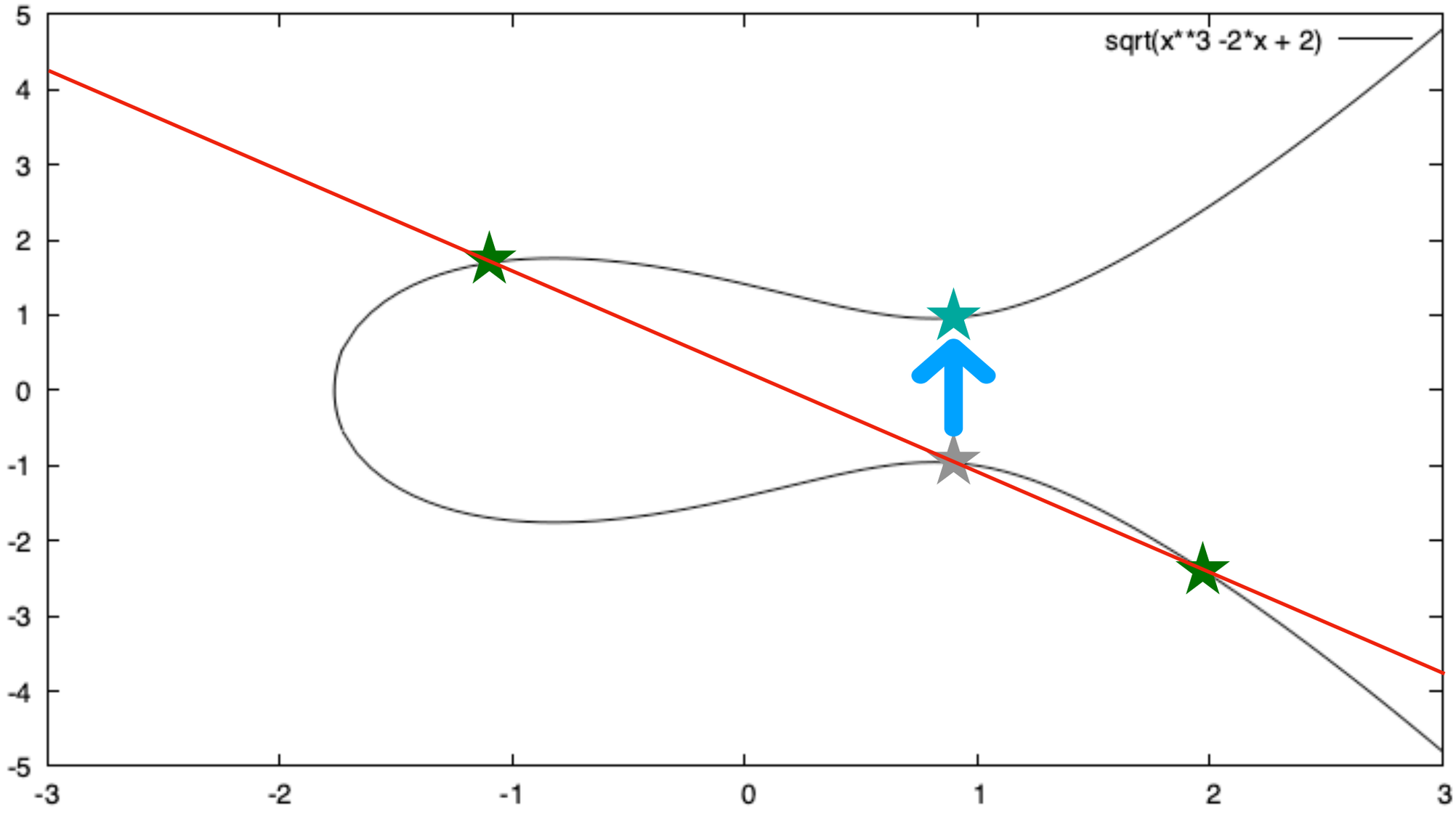
Adding points

- Given 2 points $p = (x, y)$ and $q = (u, v)$
  - Draw a line crossing those points
  - Mark the 3 point of intersection with the curve
  - Flip the point up/down



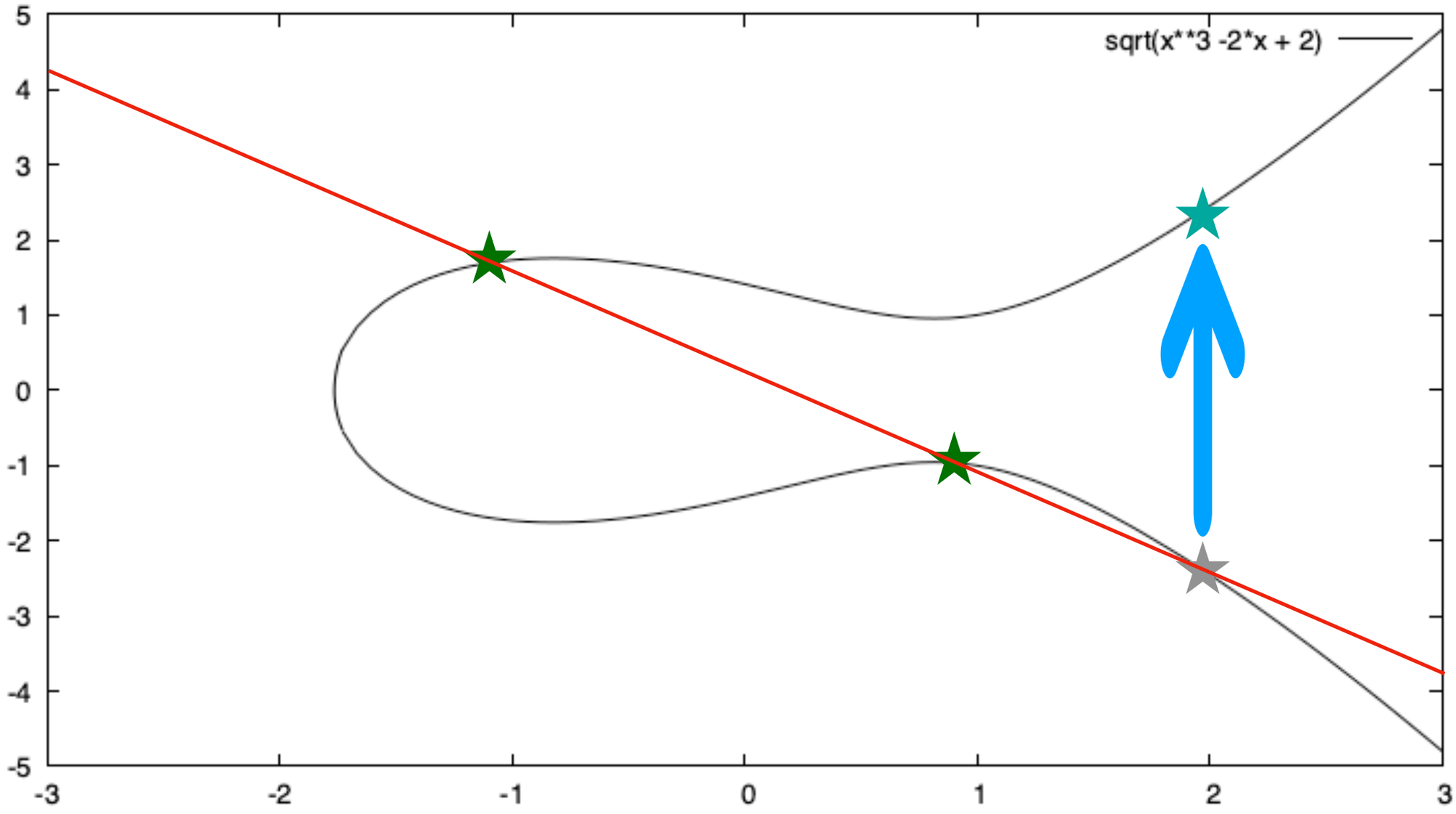
Adding points

- Given 2 points $p = (x, y)$ and $q = (u, v)$
  - Draw a line crossing those points
  - Mark the 3 point of intersection with the curve
  - Flip the point up/down



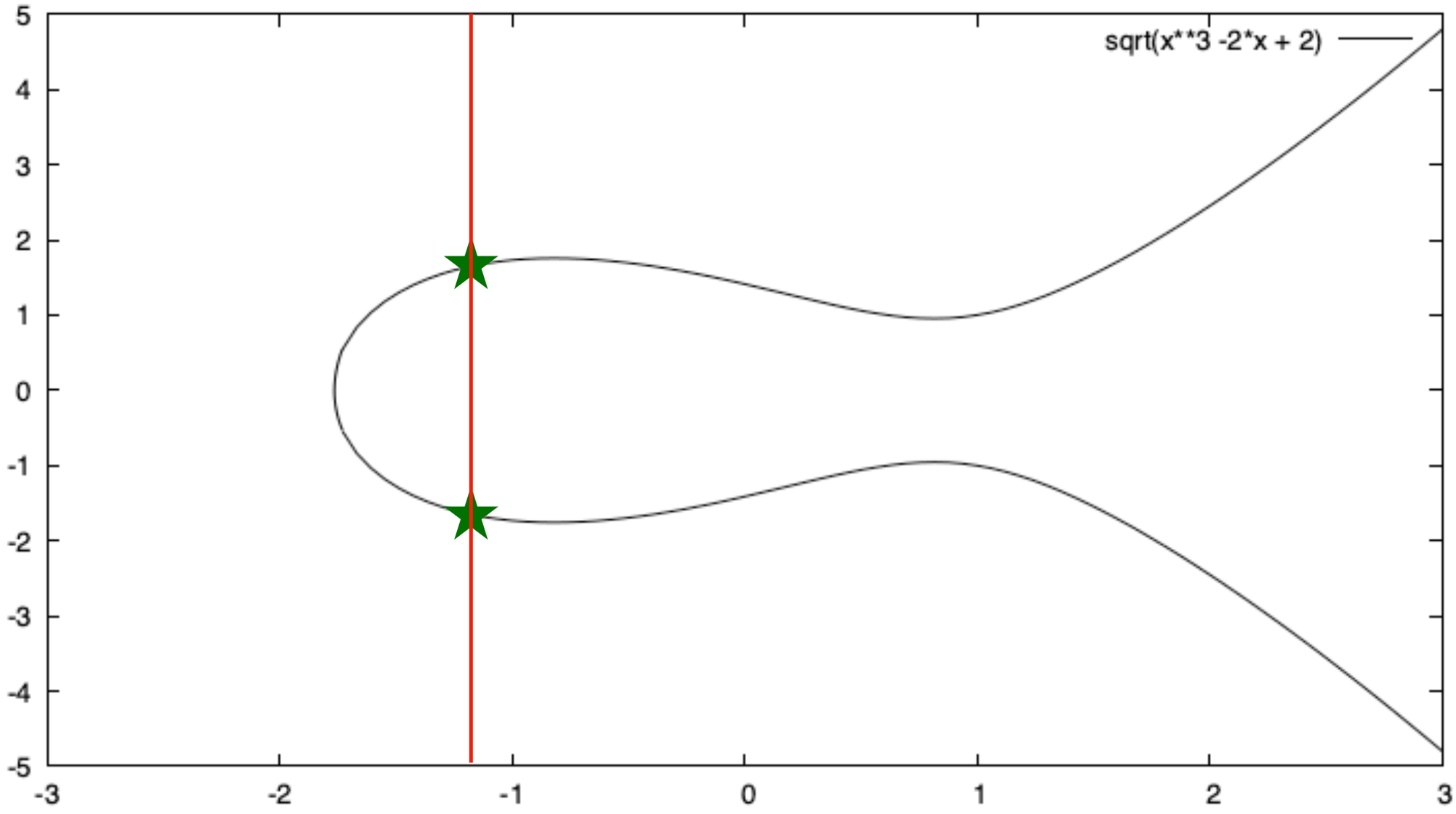
Adding points

- Given 2 points $p = (x, y)$ and $q = (u, v)$
  - Draw a line crossing those points
  - Mark the 3 point of intersection with the curve
  - Flip the point up/down



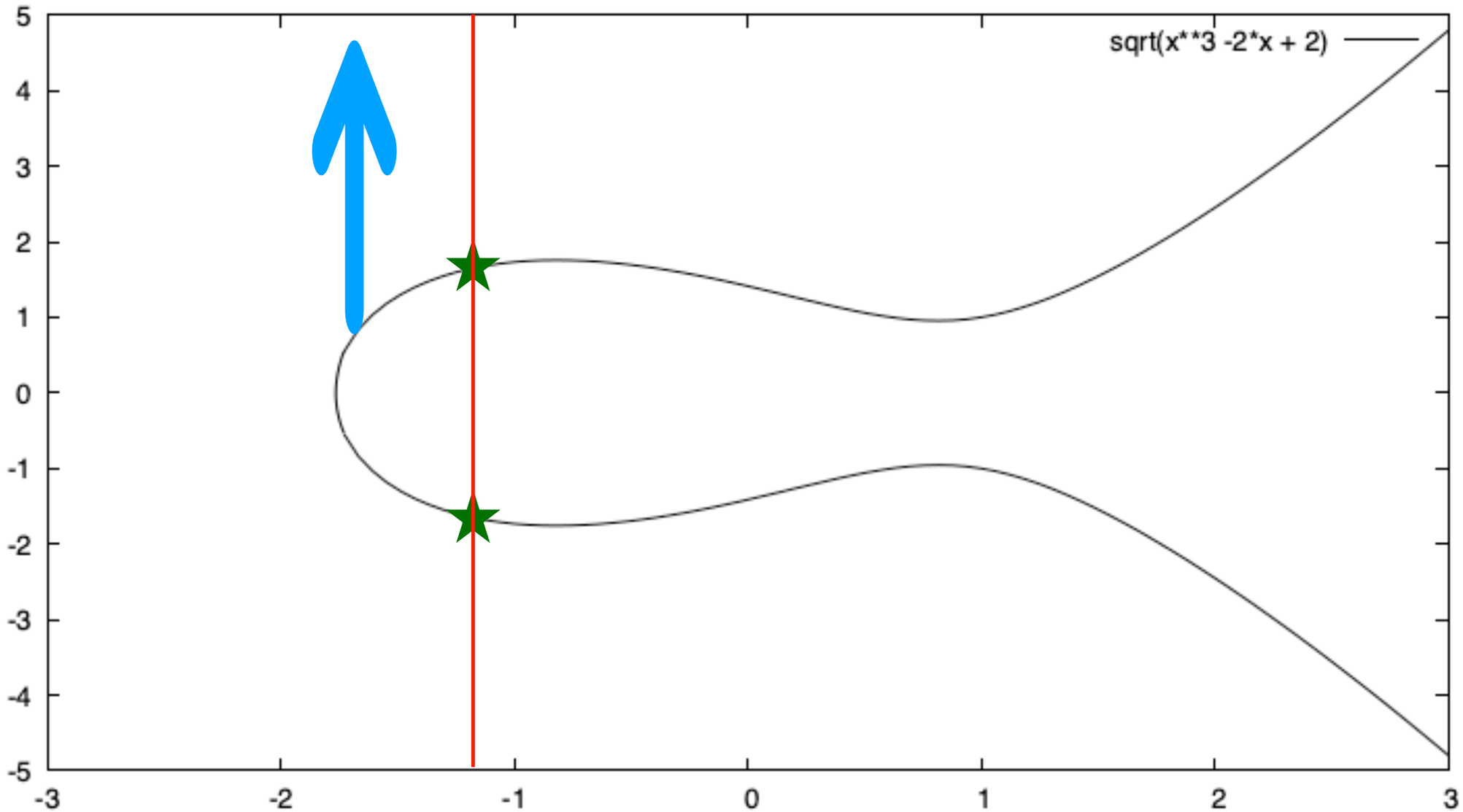
Adding points

- Definition not complete



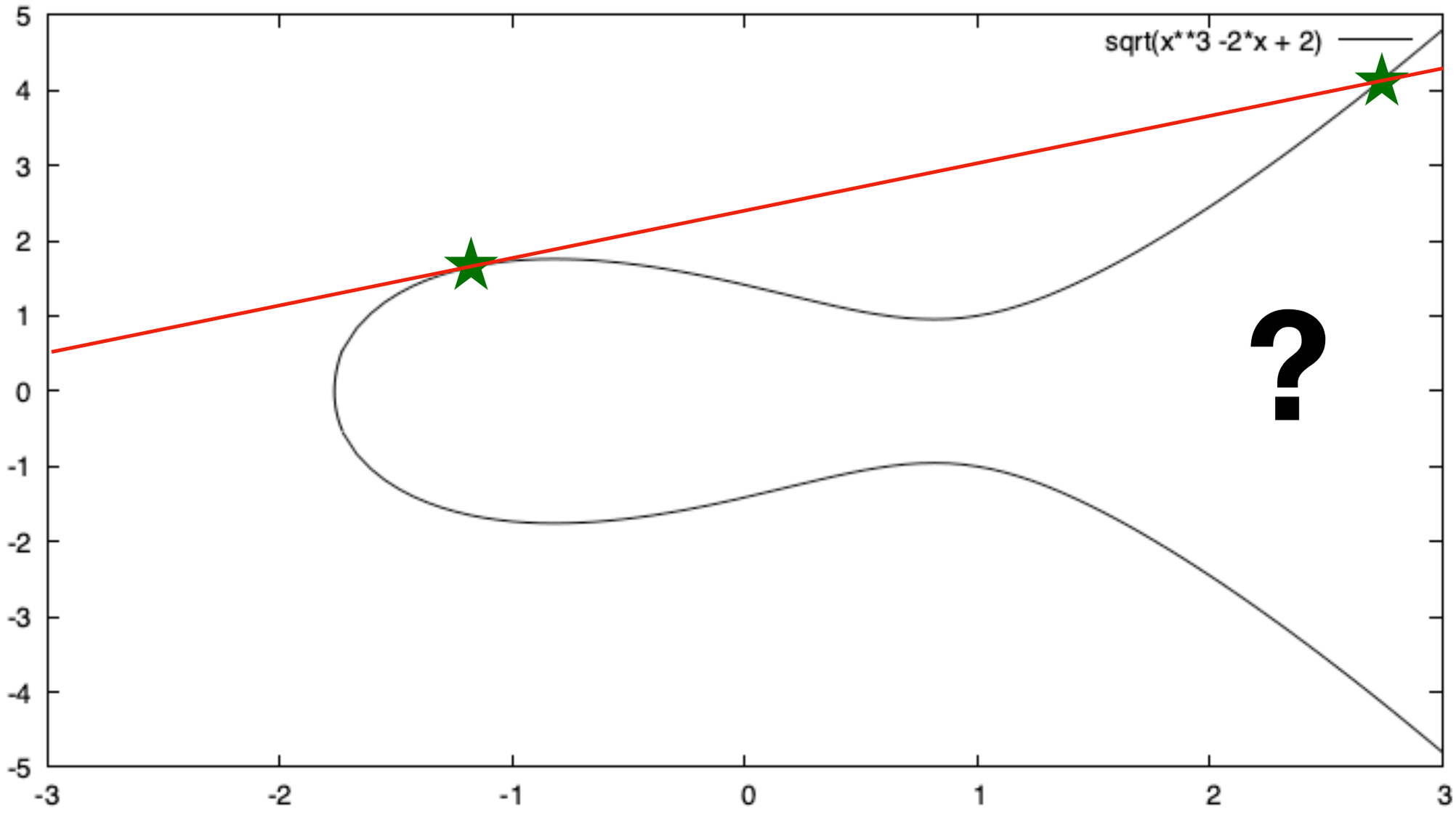
Adding points

- Definition not complete
  - => add $\pm\infty$



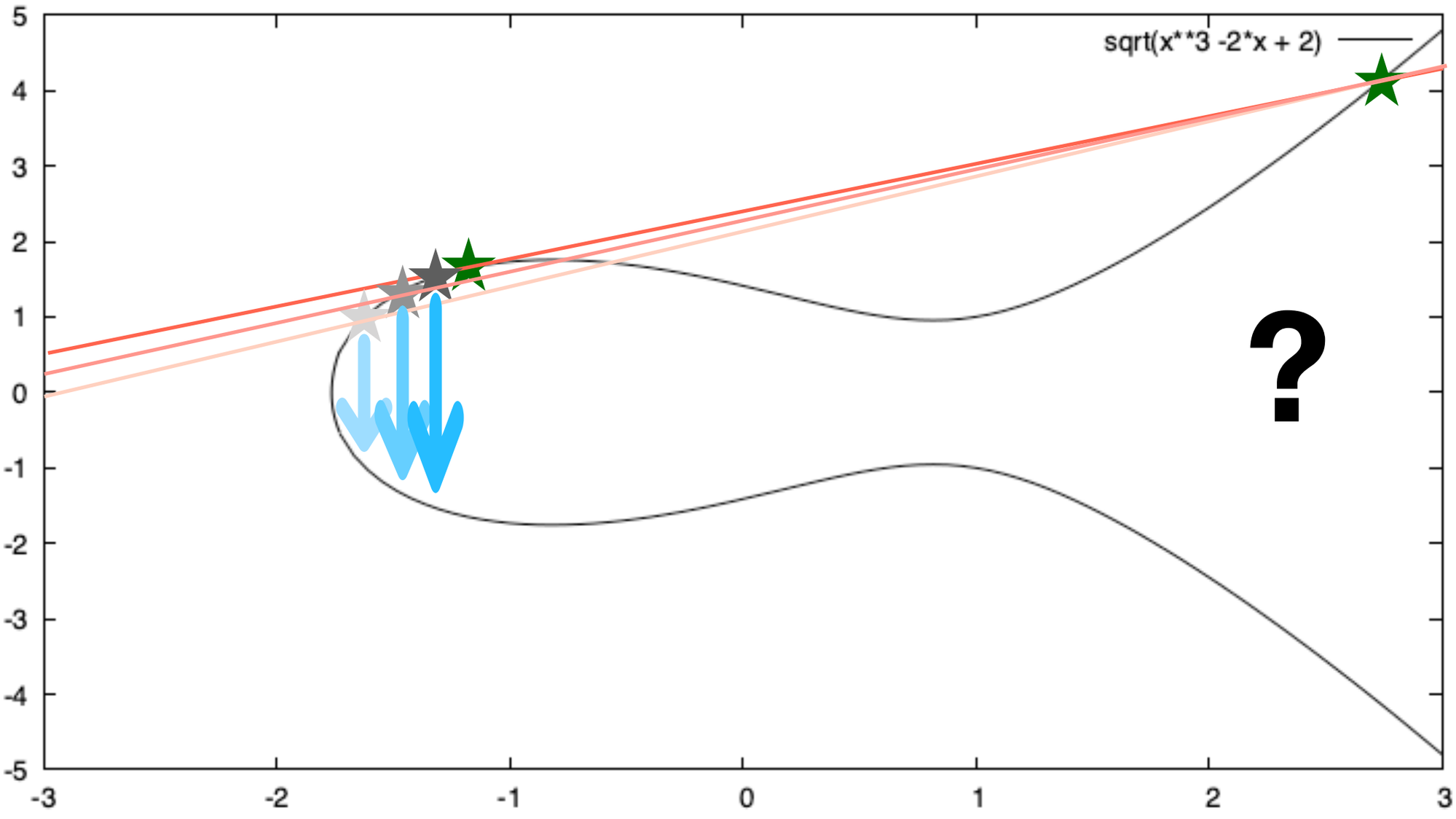
Adding points

- Definition not complete



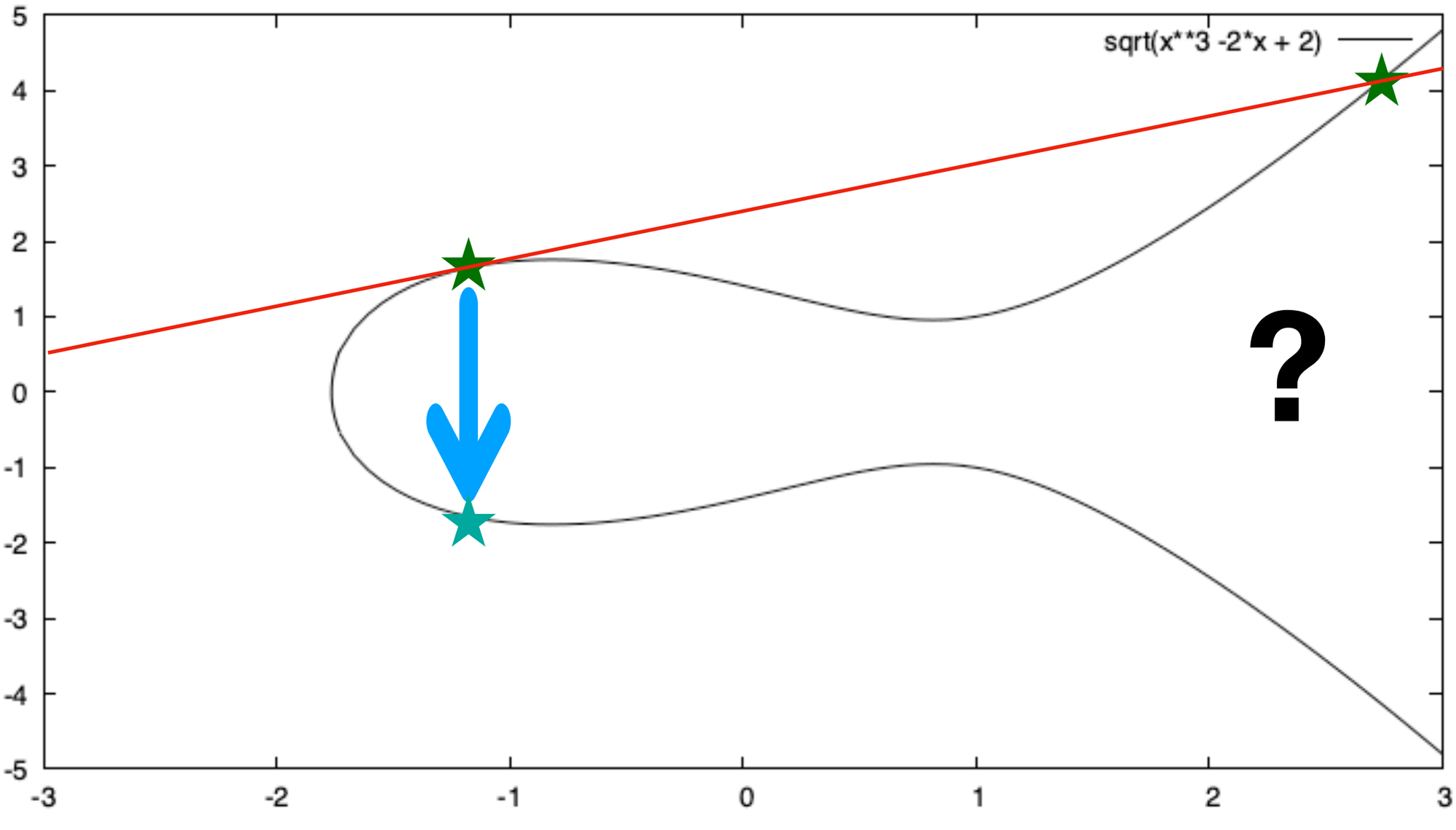
Adding points

- Definition not complete
  - Limit of getting closer and closer to that point



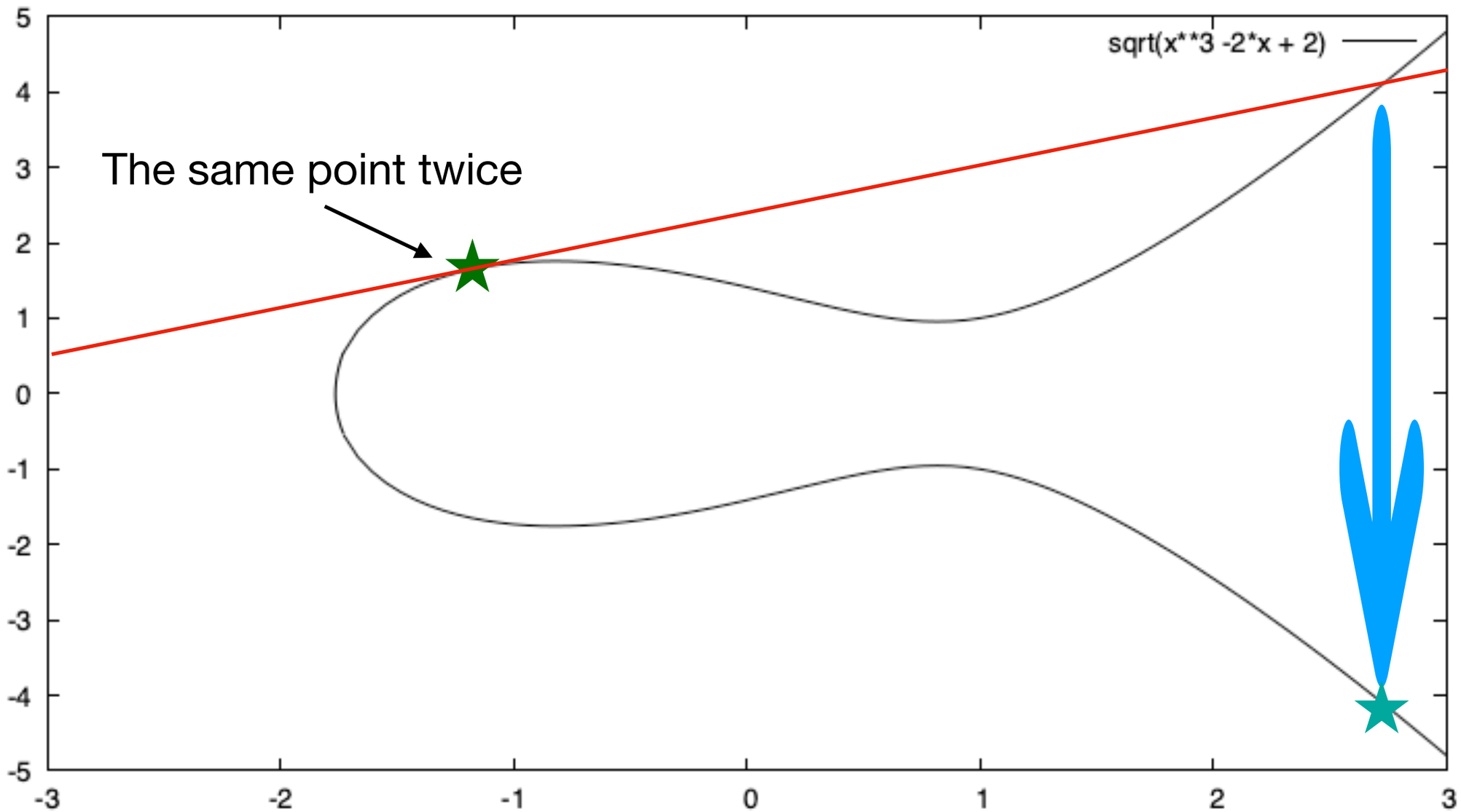
Adding points

- Definition not complete
  - Limit of getting closer and closer to that point



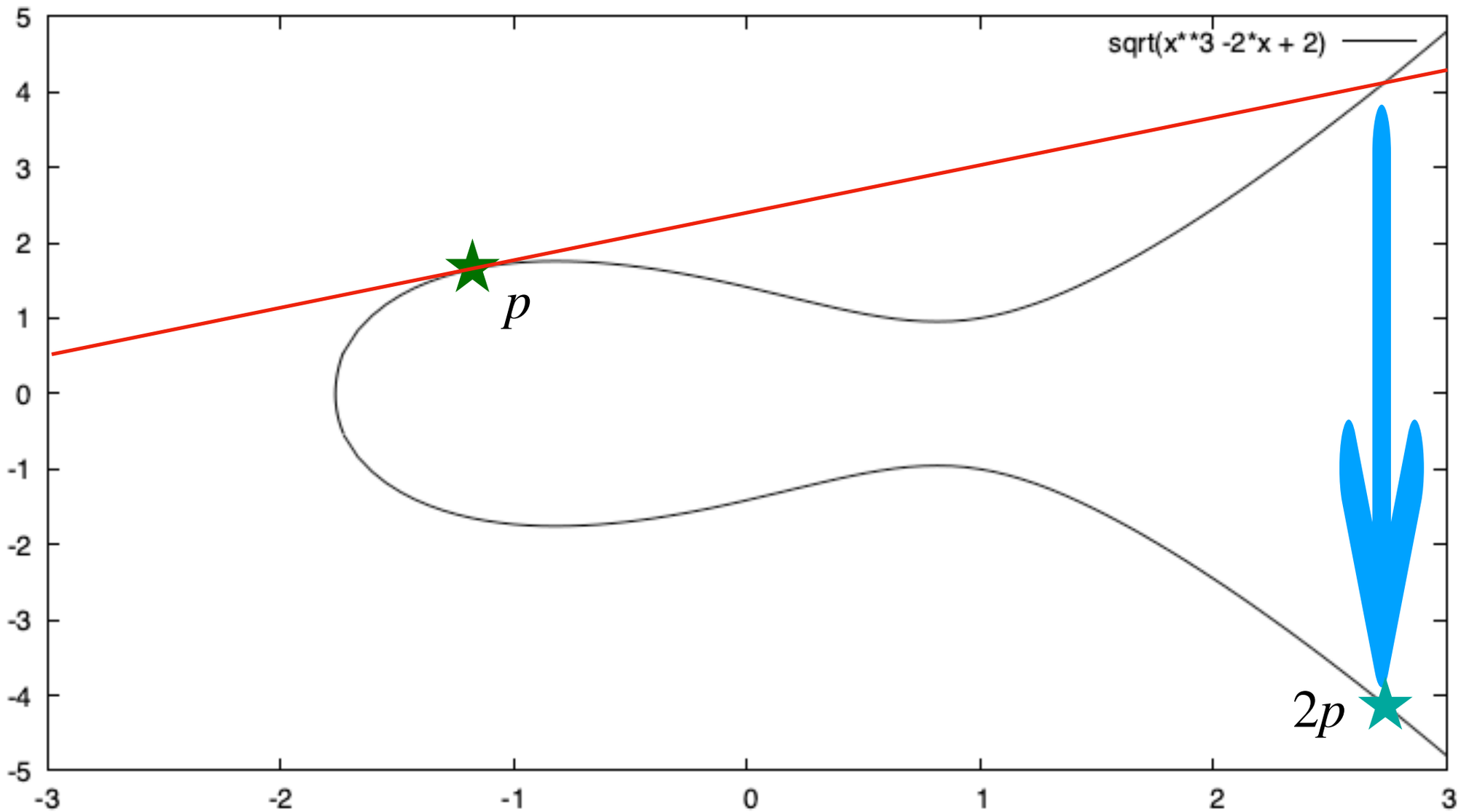
Adding points

- To add a point to itself, take the limit of 2 points getting closer and closer (the tangent of the curve at that point)



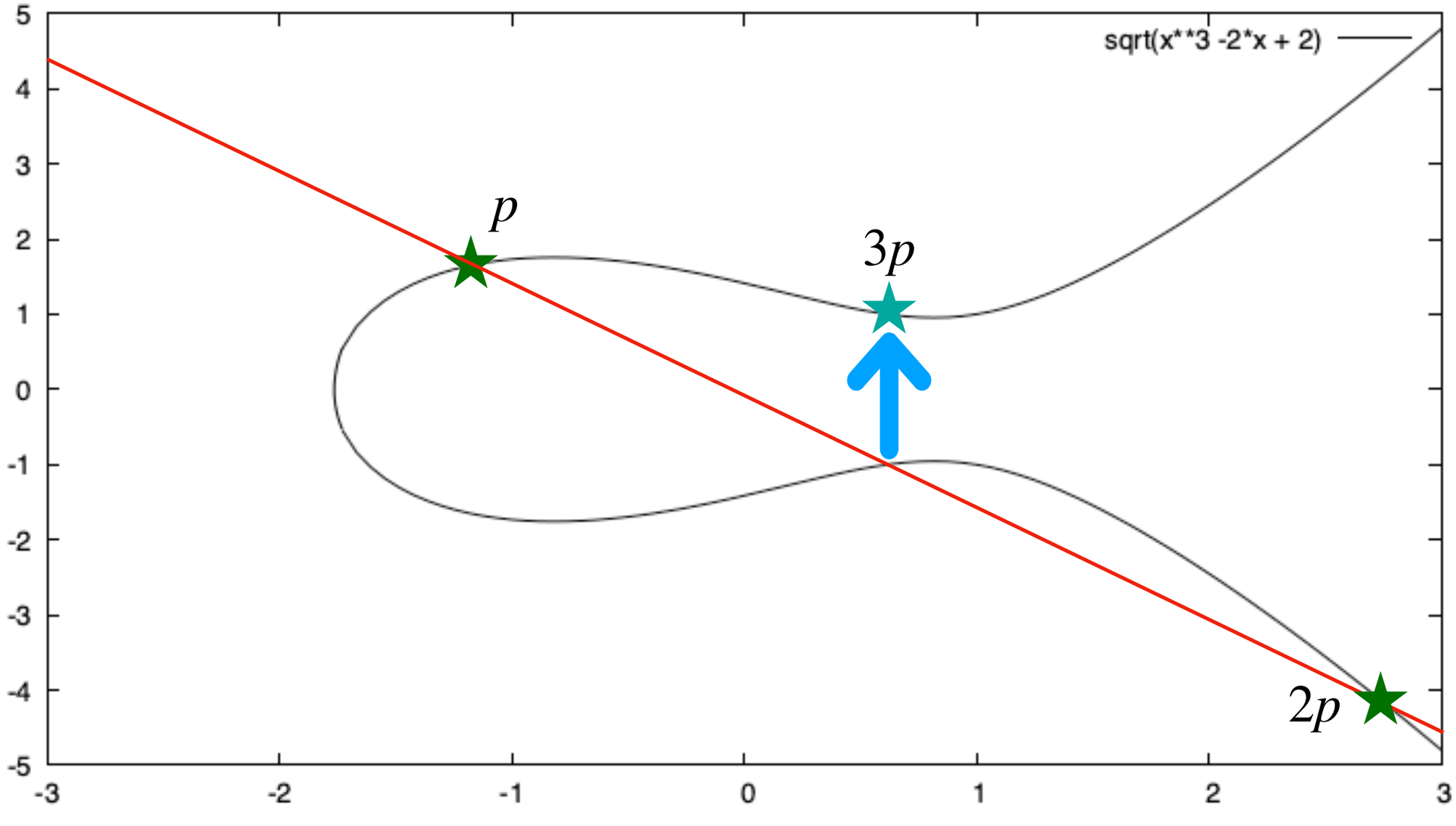
Adding points

- Let's add a point $p = (x, y)$ 2 times to itself



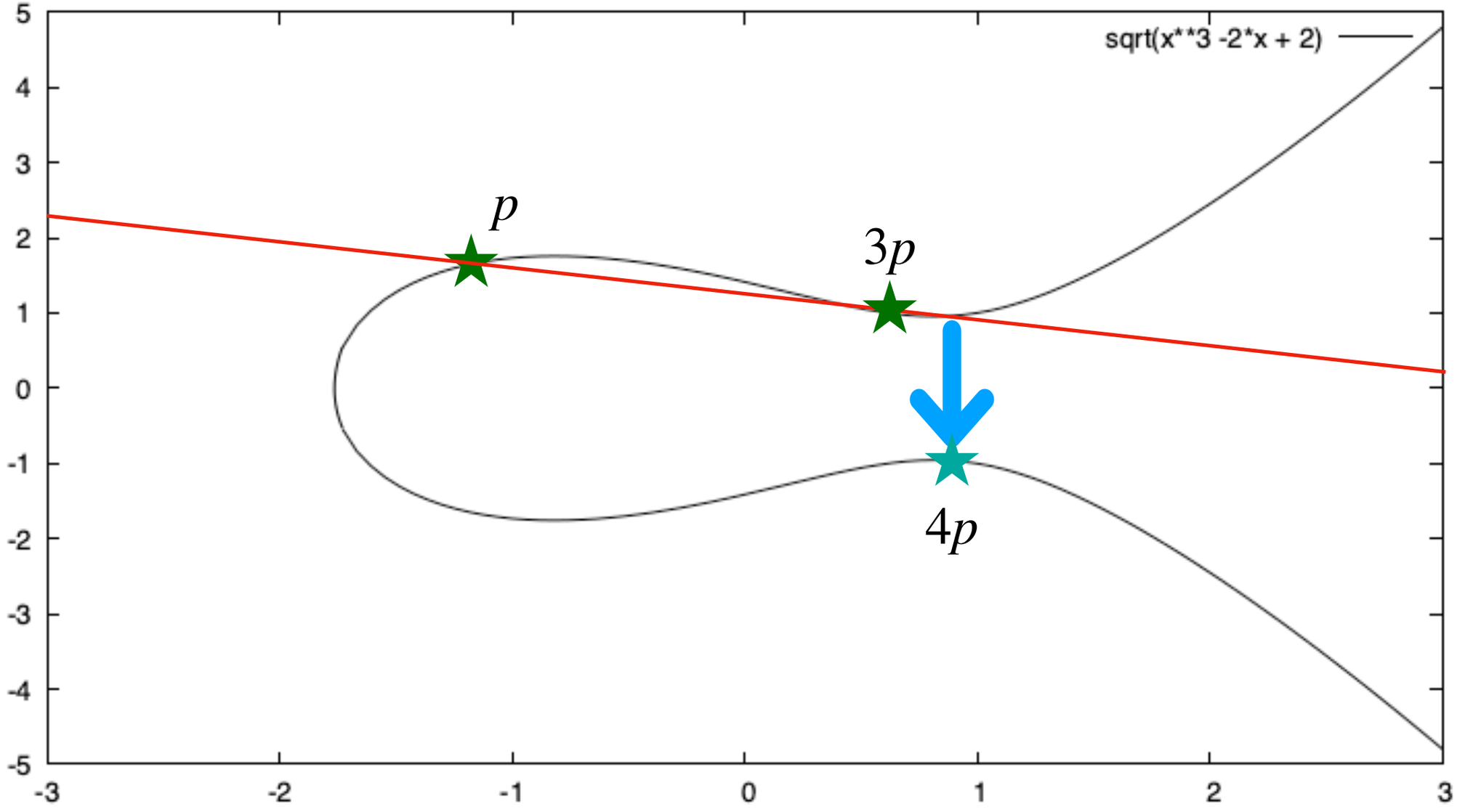
Adding points

- Let's add a point $p = (x, y)$ 3 times to itself



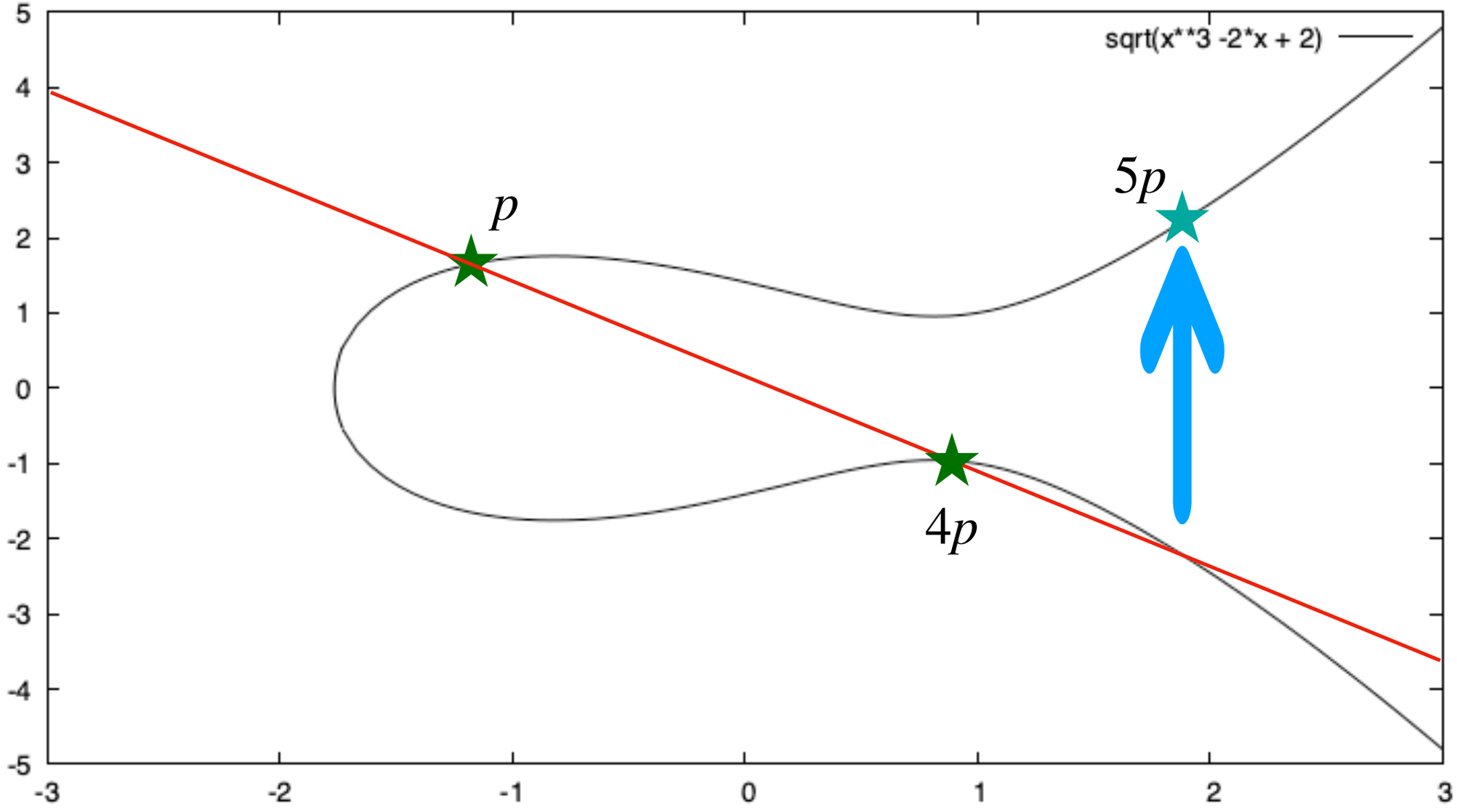
Adding points

- Let's add a point $p = (x, y)$ 4 times to itself



Adding points

- Let's add a point $p = (x, y)$ 5 times to itself

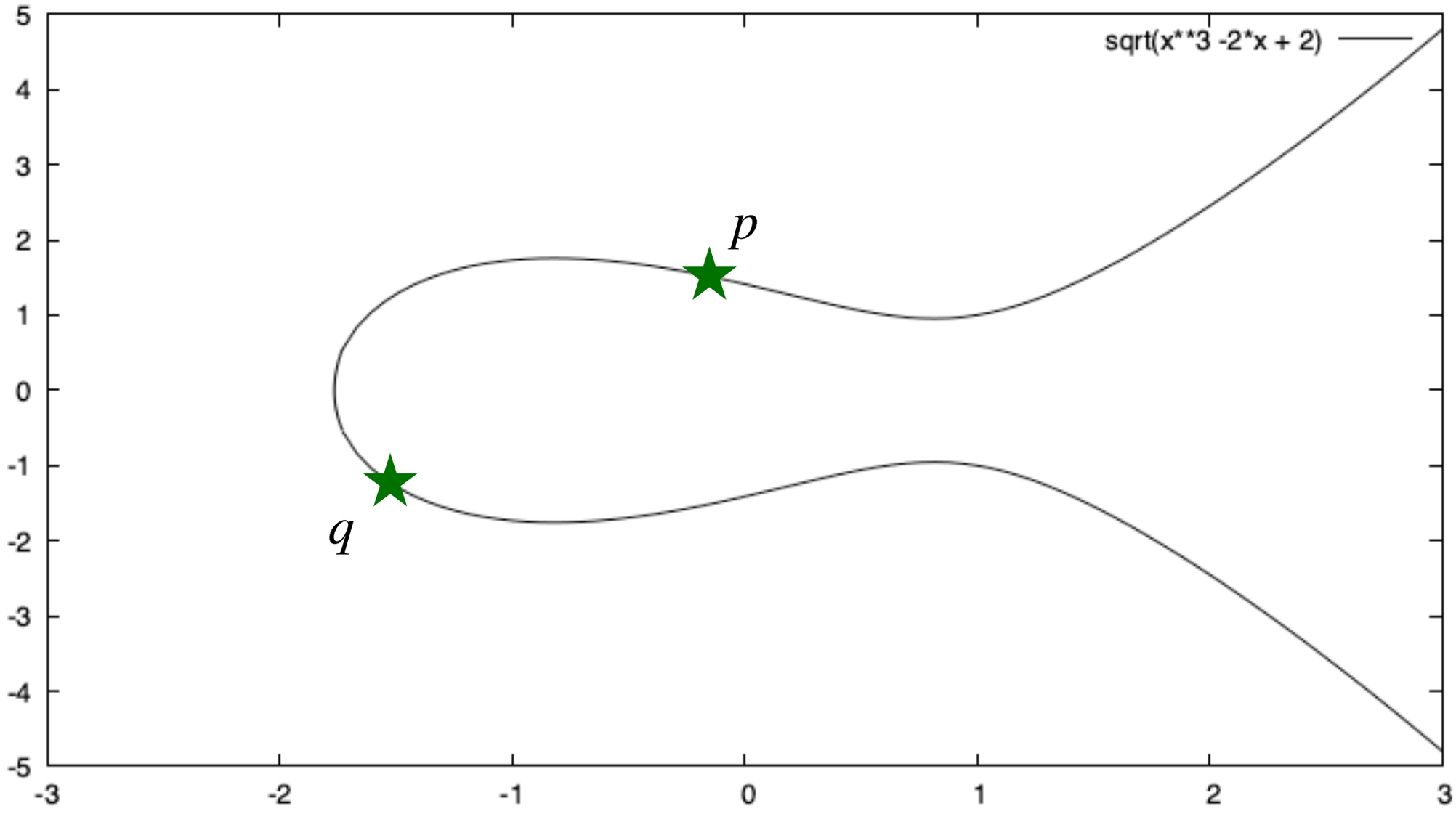


Point arithmetic on elliptic curve

- Using this definition, addition is well defined
  - P+Q = Q+P
  - P+(Q+R) = (P+Q)+R
  - P+0 = P, point 0 is the one at $\pm\infty$
    - This is the reason we “mirror” the point on the x-axis
  - -P = defined similarly

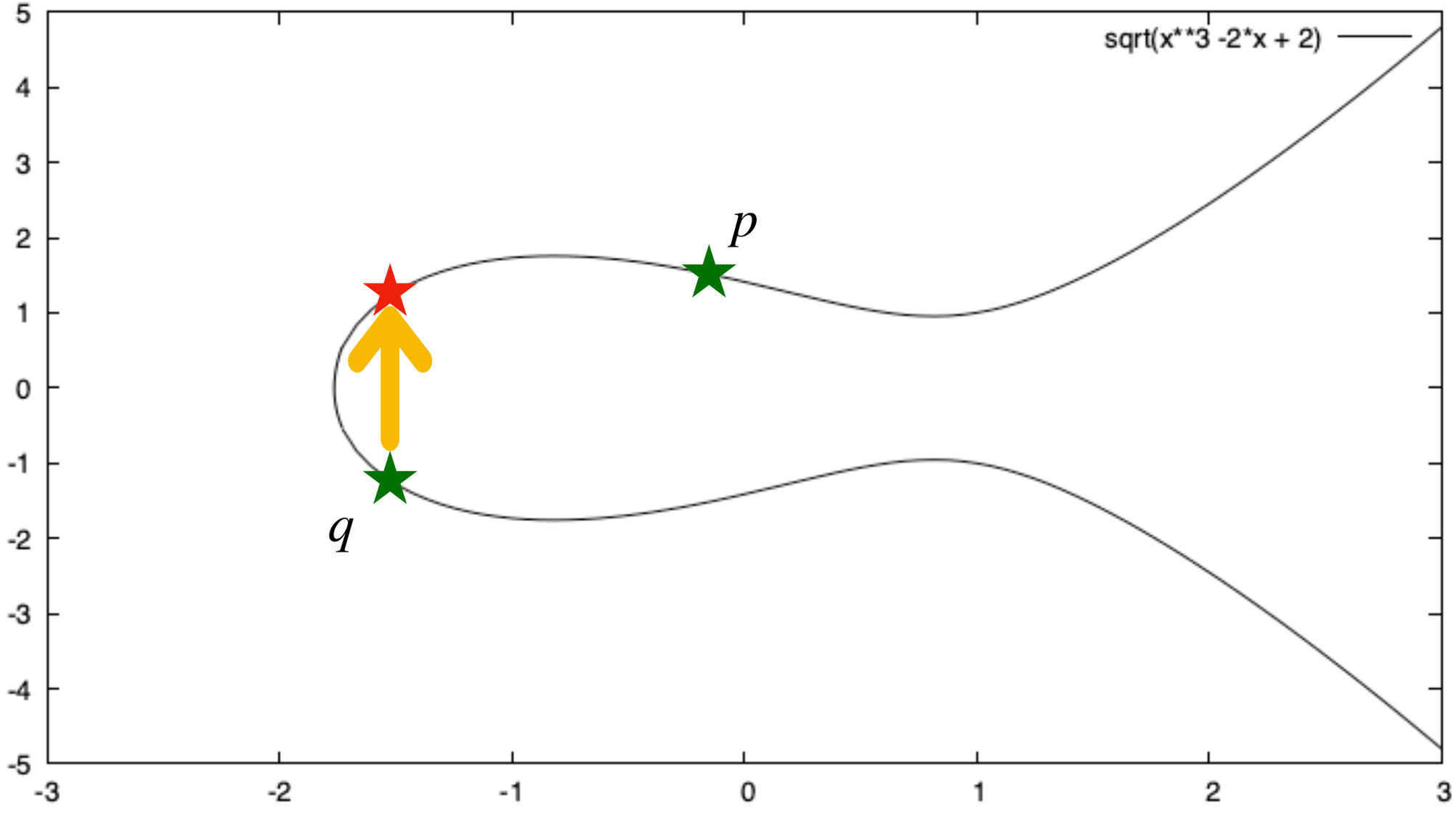
Adding points

- Point subtraction $p - q$



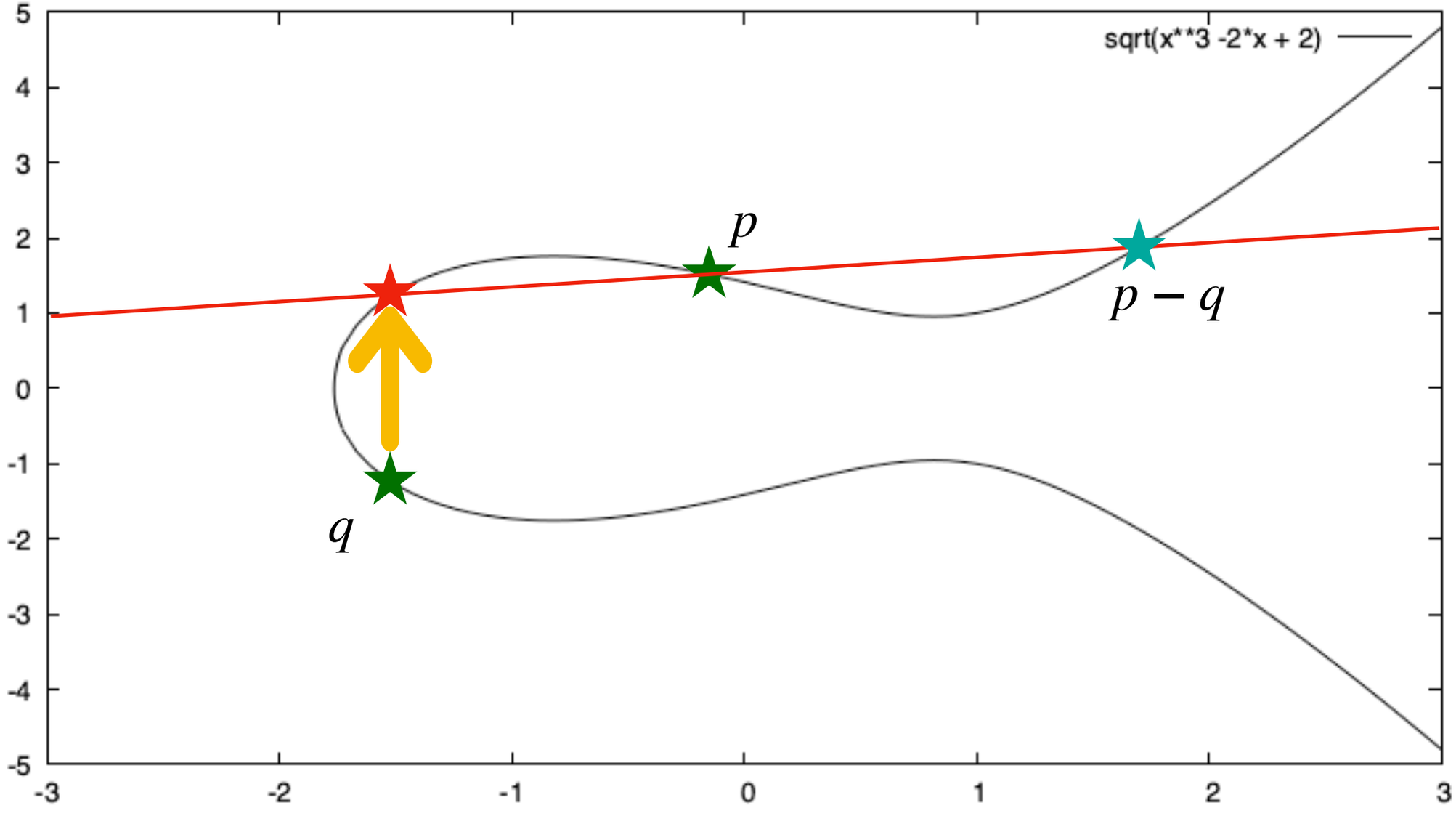
Adding points

- Point subtraction $p - q$



Adding points

- Point subtraction $p - q$



Points on elliptic curves

- The entire math of ECC is based on adding points
  - A point G can be added to itself, the new point is $H = 2G$
  - Added k times to itself results in point $F = kG$
  - Points can be added very fast

- As a side node, for ECC-based cryptography, everything ‘happens’ $\mod n$
  - A point is on a curve iff

$$y^2 \mod n = x^3 + ax + b \mod n$$

  - Point arithmetics are still well defined