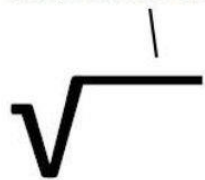


Complex Numbers

Why can't we be together?



-1
It's complex.

Imaginary Unit
Define the imaginary unit
 $i = \sqrt{-1}$ so that $i^2 = -1$

Complex Number
Any number of the form
 $a + bi$ where a and b
are real numbers.

$$\begin{aligned}x^2 + 25 &= 0 \\x^2 &= -25 \\\sqrt{x^2} &= \pm \sqrt{-25} \\x &= \pm 5i\end{aligned}$$

$$(2 - 3i)(2 + 3i)$$

$$= 4 + 6i - 6i - 9i^2$$

$$= 4 - 9(-1)$$

$$= 4 + 9$$

$$= 13$$

complex number

$$3 + \frac{1}{2}i$$

$$12 - 5i$$

$$1 - i$$

$$45i$$

$$101$$

conjugate

$$3 - \frac{1}{2}i$$

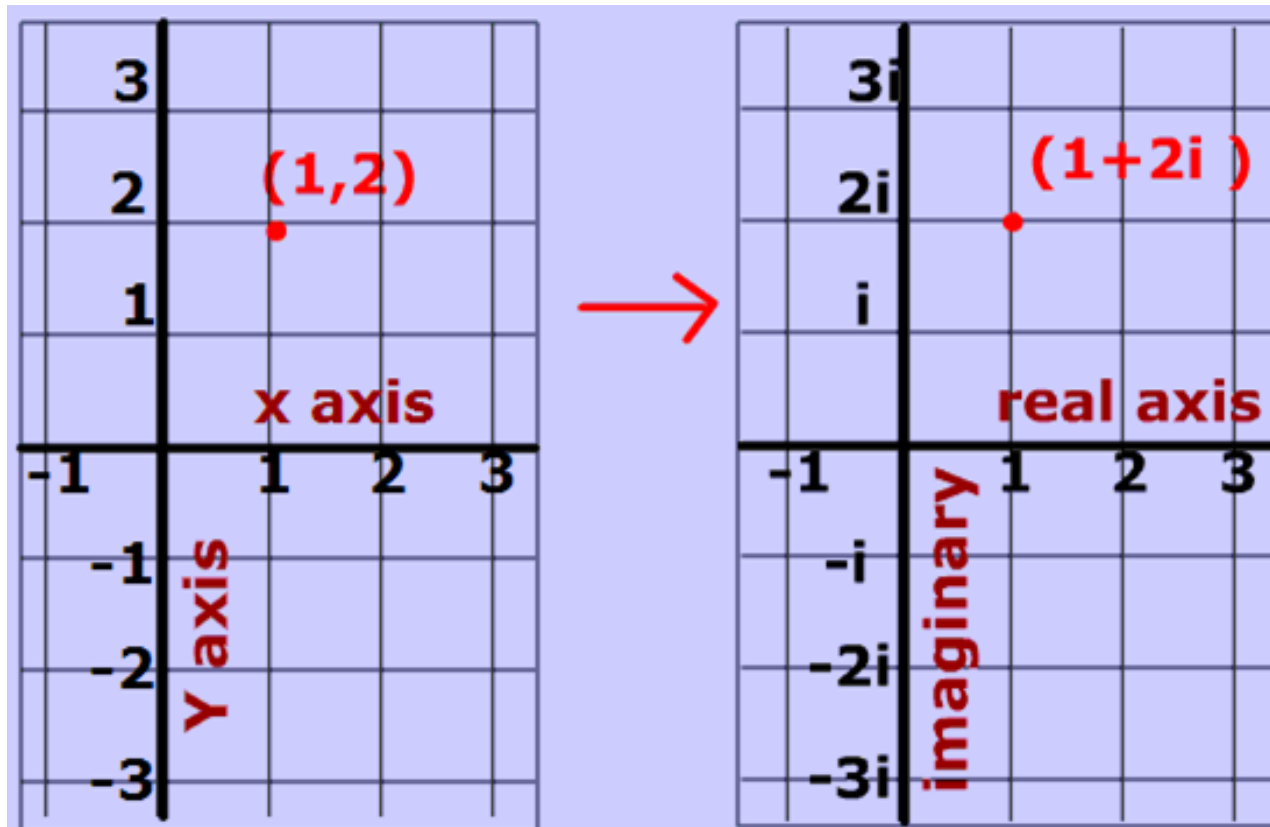
$$12 + 5i$$

$$1 + i$$

$$-45i$$

$$101$$

Complex Numbers: Cartesian



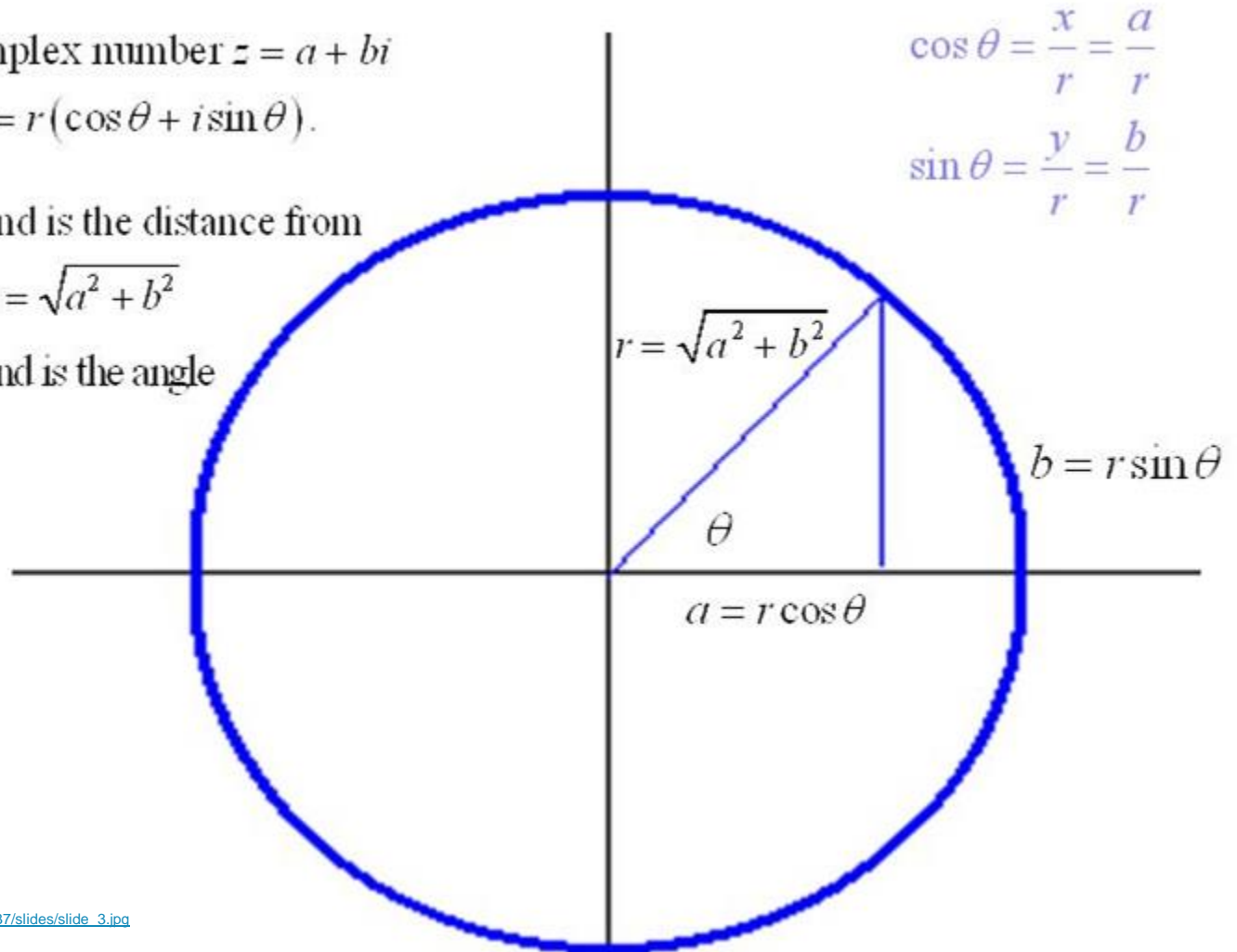
Complex Numbers: Polar

The trig form of the complex number $z = a + bi$ is $z = (r \cos \theta + ri \sin \theta) = r(\cos \theta + i \sin \theta)$.

r is called the modulus and is the distance from the origin to the point. $r = \sqrt{a^2 + b^2}$

θ is called the argument and is the angle formed with the x-axis.

$$\theta = \tan^{-1}\left(\frac{b}{a}\right)$$



Complex Numbers: Euler Formula

The polar form of a complex number can be rewritten as :

2.71828183

$$\begin{aligned} z &= r(\cos \theta + i \sin \theta) = x + iy \\ &= re^{i\theta} \end{aligned}$$

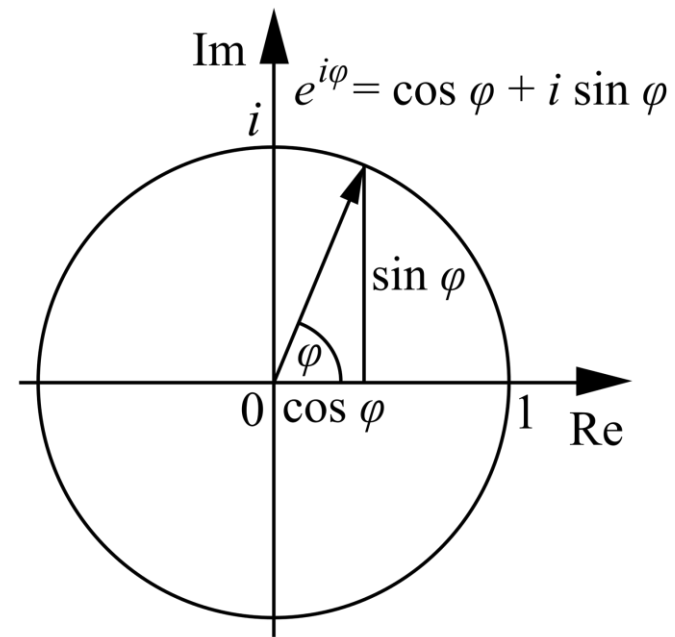
This leads to the complex exponential function :

$$\begin{aligned} e^z &= e^{x+iy} = e^x e^{iy} \\ &= e^x (\cos y + i \sin y) \end{aligned}$$

Further leads to :

$$\left\{ \begin{aligned} \cos \theta &= \frac{1}{2} (e^{j\theta} + e^{-j\theta}) \\ \sin \theta &= \frac{1}{2j} (e^{j\theta} - e^{-j\theta}) \end{aligned} \right.$$

© iTutor, 2000-2013. All Rights Reserved



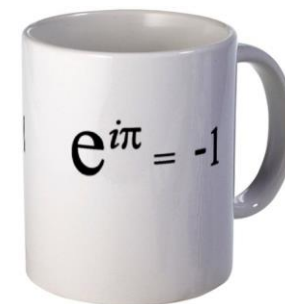
Complex Numbers: Euler Formula

$$\varphi = \frac{\pi}{2} \Rightarrow e^{i \cdot \frac{\pi}{2}} = \cos \frac{\pi}{2} + i \sin \frac{\pi}{2} \Rightarrow e^{i \cdot \frac{\pi}{2}} = 0 + 1 \cdot i = i,$$

$$\varphi = \pi \Rightarrow e^{i \cdot \pi} = \cos \pi + i \sin \pi \Rightarrow e^{i \cdot \pi} = -1 + 0 \cdot i = -1,$$

$$\varphi = \frac{3\pi}{2} \Rightarrow e^{i \cdot \frac{3\pi}{2}} = \cos \frac{3\pi}{2} + i \sin \frac{3\pi}{2} \Rightarrow e^{i \cdot \frac{3\pi}{2}} = 0 - 1 \cdot i = -i,$$

$$\varphi = 2\pi \Rightarrow e^{i \cdot 2\pi} = \cos 2\pi + i \sin 2\pi \Rightarrow e^{i \cdot 2\pi} = 1 + 0 \cdot i = 1.$$



Complex Geometric interpretation

- Complex numbers form a **vector space** over the real numbers.
- Addition, subtraction, negation and scalar multiplication are exact same in both
- Conjugation = reflect the number in the real axis
- Complex multiplication
 - Not that direct
 - But decomposes to **dot and cross product!** Wow

Complex in C++

```
#include <complex>
#include <cmath>
const double PI = acos(-1.0);

complex<double> num1(2, 3); // 2 + 3i

std::cout << num1.real() << "+" << num1.imag() << "i\n";    // 2 + 3i

complex<double> num2(1, 1);

cout << "Modulus = " << abs(num2) << "\n";           // r = 1.41421
cout << "Argument = " << arg(num2) << "\n";           // theta radian = 0.785398
cout << "Angle = " << arg(num2) * 180 / PI << "\n";     // theta degree = 45
cout << "Norm = " << norm(num2) << "\n";               // 2 (1*1 + 1*1)

complex<double> num3 = std::polar(1.41421, 0.785398);
cout << "(x+iy) from polar are: " << num3 << "\n";    // (0.999998,0.999997)

complex<double> zero;
complex<double> x_part = 15;
cout << x_part << "\n";                               // (15,0)

complex<double> a(1, 2);
complex<double> b(3, 4);

cout<<a+b<<"\n";    // (4,6)
cout<<a-b<<"\n";    // (-2,-2)
cout<<a*b<<"\n";    // (-5,10)
cout<<b*2.0<<"\n";  // (6,8)
cout<<b/2.0<<"\n";  // (1.5,2)
```

Complex in C++

```
cout << fixed << std::setprecision(1);

complex<double> i = -1;
cout<<sqrt(i)<<"\n";           // (0,1)

complex<double> c(2, 3);
cout<<conj(c)<<"\n";           // (2,-3)
cout<<pow(c, 2)<<"\n";         // (-5,12)

complex<double> i1 (0, -1);
cout<<exp(i1 * PI)<<"\n";      // (-1, 0)
```


2D point and Complex Numbers

- Complex numbers extremely fit with vectors
- We can use it as representation to (x, y)
- We can use it for operations $(+, -, /, *$ scalar)
- We can do work around for dot/cross products
- We can have some polar representation and operations over it

Complex Numbers: Point Basics

```
#include <complex>
const double PI = acos(-1.0);
const double EPS = (1e-10);
typedef complex<double> point;

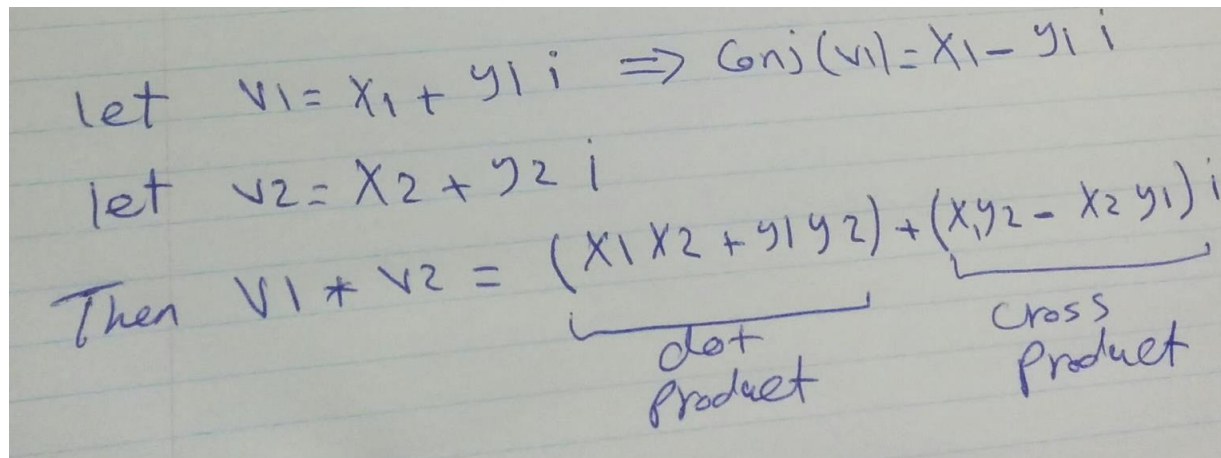
#define X real()
#define Y imag()

// We can rewrite any of following as functions
#define angle(a) (atan2((a).imag(), (a).real()))
#define vec(a,b) ((b)-(a))

#define length(a) (hypot((a).imag(), (a).real()))
#define normalize(a) (a)/length(a)
```

- Avoid sqrt, use hypot, it **avoids overflow**
- However, hypot is **slow** function
- Don't compute hypot with actual distance if **squared** distance is enough. Use dot product

Complex Numbers: Dot/Cross



Handwritten derivation showing the multiplication of two complex numbers $v_1 = x_1 + y_1 i$ and $v_2 = x_2 + y_2 i$. The result is expressed as the sum of a dot product and a cross product.

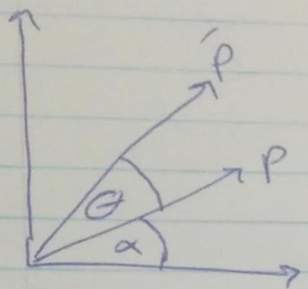
$$\begin{aligned} \text{let } v_1 &= x_1 + y_1 i \Rightarrow \text{conj}(v_1) = x_1 - y_1 i \\ \text{let } v_2 &= x_2 + y_2 i \\ \text{Then } v_1 * v_2 &= \underbrace{(x_1 x_2 + y_1 y_2)}_{\text{dot product}} + \underbrace{(x_1 y_2 - x_2 y_1) i}_{\text{cross product}} \end{aligned}$$

```
// dp = a*b cos(T), if zero -> prep
#define dp(a,b) ((conj(a)*(b)).real() )
// cp = a*b sin(T), if zero -> parallel
#define cp(a,b) ((conj(a)*(b)).imag() )
#define same(p1,p2) (dp(vec(p1,p2),vec(p1,p2)) < EPS)
#define lengthSqr(p) dp(p,p)
```

- Avoid using norm(x). Use dp(x, x) instead

Complex Numbers: Rotation

Let's rotate point P with angle Θ
let $P = r e^{i\alpha}$ \Rightarrow rotation just increase angle



\hat{P} has same r
but angle $= \alpha + \Theta$

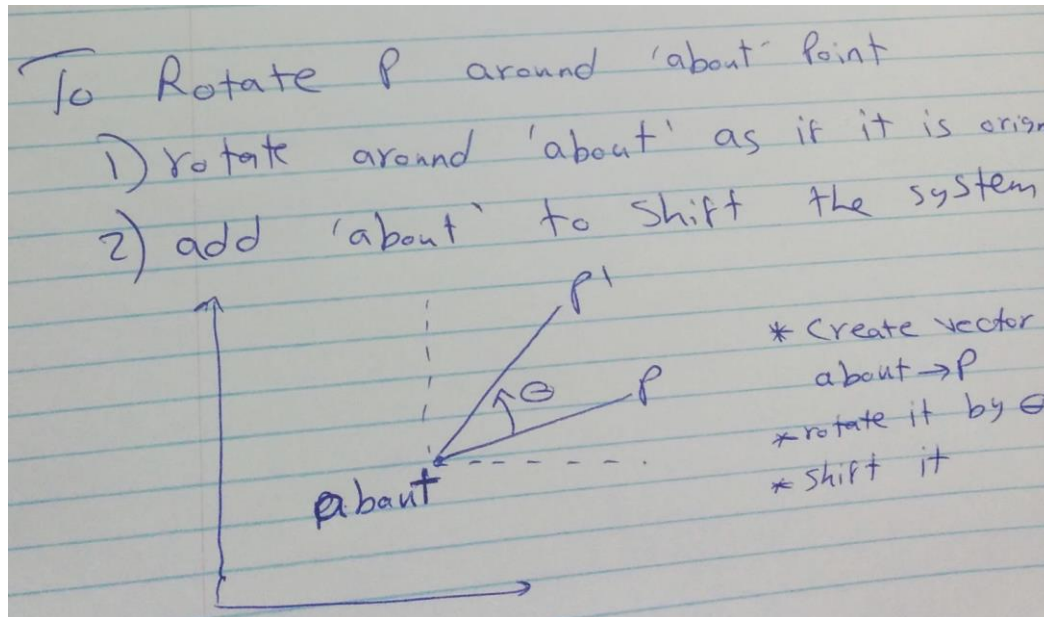
then $P * e^{i\Theta} = r * e^{i(\alpha + \Theta)}$
 \hookrightarrow c++: `exp(Point(0, \Theta))`

This is rotation around origin

```
#define rotate0(p,ang)
```

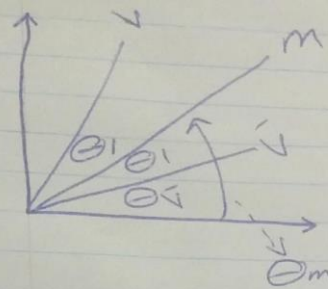
```
((p)*exp(point(0,ang)))
```

Complex Numbers: Rotation



```
#define rotateA(p,ang,about) (rotate0(vec(about,p),ang)+about)
```

Complex Numbers: Reflection



reflect (mirror) vector v around m to \hat{v}

$$\theta_m = \theta_1 + \theta_{\hat{v}}$$

$$\theta_v = \theta_1 + \theta_m$$

Re-arrange

$$\theta_{\hat{v}} = \theta_m - (\theta_v - \theta_m)$$

$$\boxed{\theta_{\hat{v}} = 2\theta_m - \theta_v}$$

let $m = x e^{i\theta_m}$
 $v = r e^{i\theta_v}$

evaluate $\boxed{\text{conj}(\frac{v}{m}) * m}$

$$\frac{v}{m} = \frac{r e^{i\theta_v}}{x e^{i\theta_m}} = \frac{r}{x} e^{i\theta_v - i\theta_m} \Rightarrow \text{conj} \Rightarrow \frac{r}{x} e^{i\theta_m - i\theta_v}$$

$$\boxed{r e^{i(2\theta_m - \theta_v)}} = m = x e^{i\theta_m} *$$

```
#define reflect0(v,m) (conj((v)/(m))*(m))
```

Complex Numbers: Reflection

```
// Refelect point p1 around p0p1
point reflect(point p, point p0, point p1) {
    point z = p-p0, w = p1-p0;
    return conj(z/w)*w + p0;
}
```