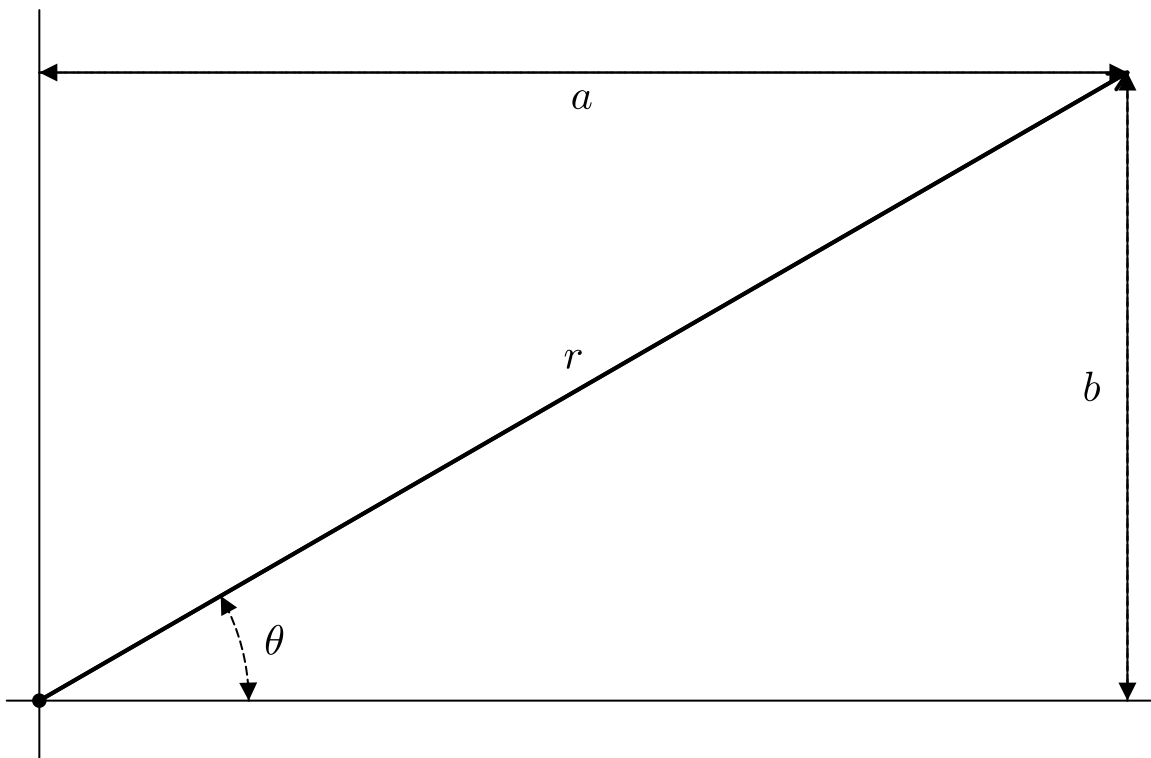


Les nombres complexes ⇄

- Introduction aux complexes [AE18; Appendix I]
- Introduction aux polynômes [AE18; P.6]

[AE18] R. A. Adams and C. Essex. *Calculus : A Complete Course*. 9 Edition (Pearson, 2018).

Cartésienne	Polaire	Exponentielle
$z = a + bi$	$z = r \operatorname{cis}(\theta)$	$z = r \exp(i\theta) = re^{i\theta}$
$a = r \cos(\theta)$	$r = \sqrt{a^2 + b^2}$	
$b = r \sin(\theta)$	$\theta = \tan^{-1}(b/a)$	



Cartésienne : facile pour addition, soustraction

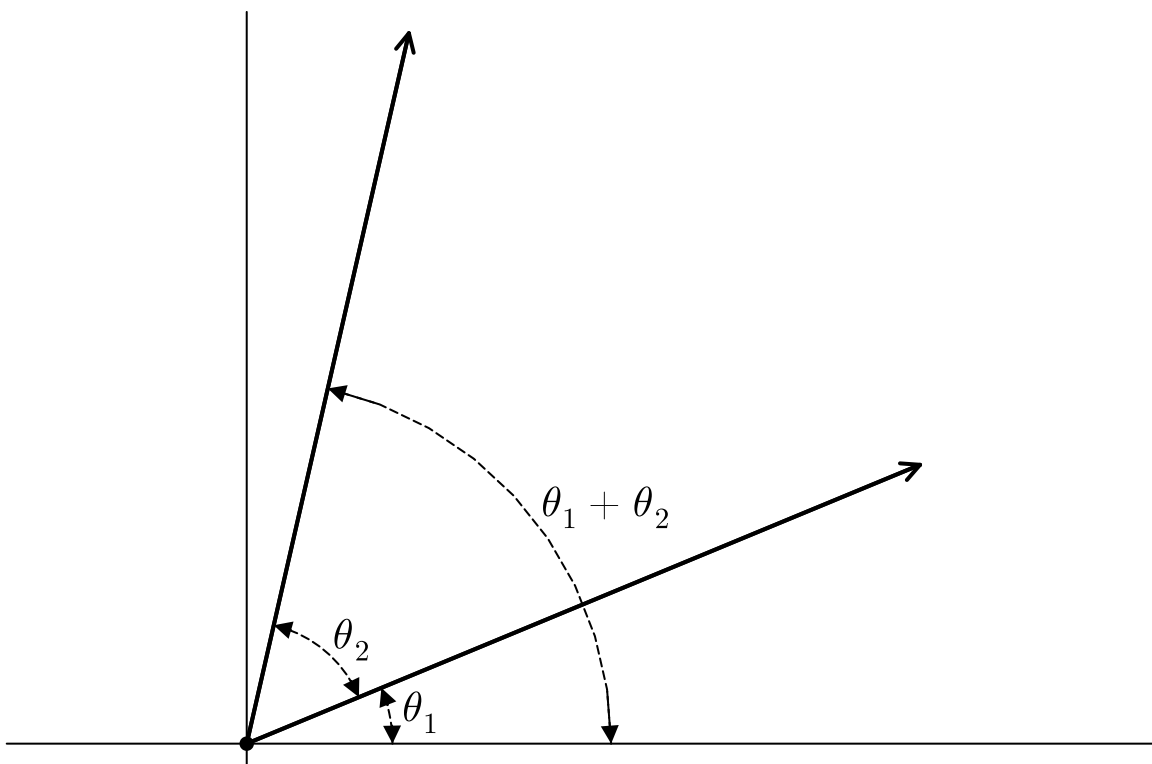
$$(a_1 + b_1i) \pm (a_2 + b_2i) = (a_1 \pm a_2) + (b_1 \pm b_2)i$$

Polaire : facile pour multiplier, diviser. Géométriquement : **rotation** d'angle θ

$$r_1 e^{i\theta_1} r_2 e^{i\theta_2} = r_1 r_2 e^{i(\theta_1 + \theta_2)}$$

$$\frac{r_1 e^{i\theta_1}}{r_2 e^{i\theta_2}} = \frac{r_1}{r_2} e^{i(\theta_1 - \theta_2)}$$

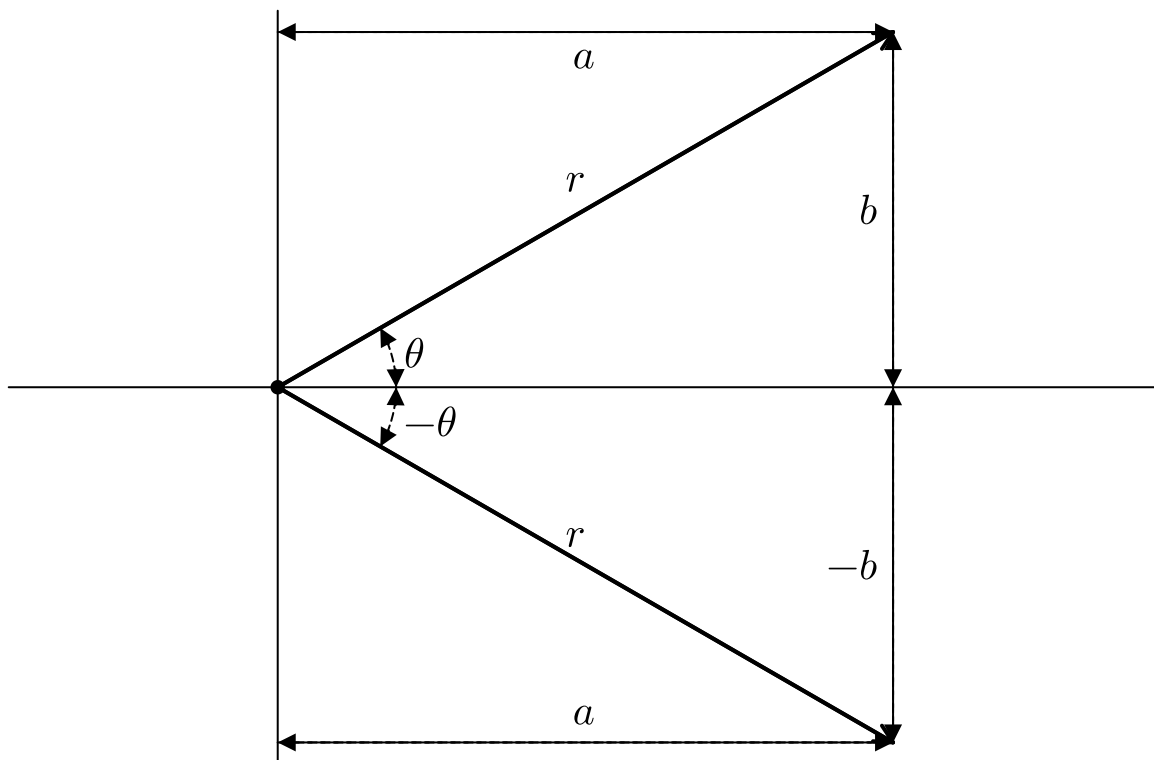
$$(r e^{i\theta})^{-1} = r^{-1} e^{-i\theta}$$



Définition : *Complexe conjugué*

$$z = a + bi \quad \bar{z} = a - bi$$

$$z = r e^{i\theta} \quad \bar{z} = r e^{-i\theta}$$



Quiz : Quel est la valeur $z\bar{z}$? (vous pouvez choisir entre cartésienne ou polaire)

► Réponse

Le théorème fondamental de l'algèbre ⇔

Tout polynôme $a_0 + a_1x + a_2x^2 + \dots + a_nx^n$ de degré n a n racines complexes (ce qui n'exclut pas que certaines d'entre elles aient une partie imaginaire nulle et donc soient réelles). Dans cette liste de n racines, certaines apparaissent plusieurs fois. Mais qu'est-ce que ça signifie qu'une racine soit racine plusieurs fois ?

Multiplicité ⇔

Une valeur z est une racine du polynôme $P(x)$ si $P(z) = 0$. Si z est également une racine de la dérivée, c'est à dire $P'(z) = 0$ alors z est une racine double. En général, z est une racine de multiplicité m si $P(z) = P'(z) = \dots = P^{(m)}(z) = 0$ où $P^{(k)}$ est la dérivée m ième de P .

De façon équivalente, ça revient à l'exposant du facteur $(x - z)$ dans la factorisation du polynôme. Par exemple, si :

$$P(x) = (x - z)^m \cdot Q(x),$$

où $Q(z) \neq 0$, alors z est une racine de multiplicité m .

Propriété des paires conjuguées ⇔

Si $z = a + ib$ est une racine de multiplicité m , alors la racine conjuguée $\bar{z} = a - ib$ possède également une multiplicité m .

► Preuve

Le cas quadratique ⇔

$$ax^2 + bx + c = 0 \quad \Rightarrow \quad x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

► Preuve

Les racines de 1 ⇔

Exemple introductif : les racines cubiques de 1 ⇔

Quelles sont les solutions de $x^3 = 1$. En d'autre termes, quels sont les racines de $x^3 - 1$.

```
cube = -1 + x3
```

```
1 cube = Polynomial([-1, 0, 0, 1])
```

On sait deviner une première racine : 1 car $1^3 = 1$.

```
première_racine = -1 + x
```

```
1 première_racine = Polynomial([-1, 1])
```

Grace à Horner, on sait factoriser cette racine, ce qui donne

$$x^3 - 1 = (x - 1)(x^2 + x + 1)$$

```
q = 1.0 + 1.0·x + 1.0·x2
```

```
1 q = div(cube, première_racine)
```

$$-1.0 + 1.0 \cdot x^3$$

1 `première_racine * q`

On trouve alors les deux racines manquantes à l'aide des formules du cas quadratique

$$\Delta = -3$$

1 `Δ = 1 - 4`

► `(-0.5+0.866025im, -0.5-0.866025im)`

1 `(-1 + √(abs(Δ)) * im) / 2, (-1 - √(abs(Δ)) * im) / 2`

Le cas général ⇔

Pour calculer les valeurs propres d'un polynôme de haut degré, il n'y a plus de formule. Mais on sait les calculer en reformulant le problème en problème de calcul de valeurs propre de la matrice companion. Comme on sait calculer numériquement les valeurs propres d'une matrices, ça nous permet de de calculer numériquement les valeurs propres d'un polynôme.

Des valeurs propres aux racines d'un polynômes ⇔

Une matrice A a comme valeur propre λ avec vecteur propre associé x (non-nul) si

$$Ax = \lambda x$$

ou de façon équivalente

$$(A - \lambda I)x = 0$$

Si la matrice $A - \lambda I$ était régulière, il n'y aurait qu'une seule solution : $x = 0$. Seulement, le fait que λ soit une valeur propre est équivalente au fait qu'il existe une solution x non-nulle. Ceci équivaut à dire que le déterminant:

$$\det(A - \lambda I) = 0$$

Ce déterminant est un polynôme en λ de degré n .

Des racines d'un polynômes aux valeurs propres ⇔

Comment trouver les n racines ? Commençons par un exemple:

► `(-1 + x, -2 + x, -3 + x)`

1 `p1, p2, p3 = Polynomial([-1, 1]), Polynomial([-2, 1]), Polynomial([-3, 1])`

$$p = -6 + 11 \cdot x - 6 \cdot x^2 + x^3$$

$$1 \quad p = p1 * p2 * p3$$

► [1.0, 2.0, 3.0]

$$1 \quad \text{roots}(p)$$

Comment est-ce que roots a retrouvé les racines à partir des coefficients ?

Commençons par essayer de trouver une des racines. Comment trouver r ?

$$(x - r)q = \alpha p$$

Où α est une constante et le polynôme q est de la forme

$$q = b_2 x^2 + b_1 x + b_0$$

En comparant les termes en x^3 , on trouve que $\alpha = b_2$. On a donc:

$$xq - b_2 p = r q$$

$$b_1 x^2 + b_0 x - b_2(-6x^2 + 11x - 6) = r(b_2 x^2 + b_1 x + b_0)$$

En comparant les coefficients des monômes 1, x et x^2 , on a 3 équations:

$$\begin{bmatrix} 0 & 0 & 6 \\ 1 & 0 & -11 \\ 0 & 1 & 6 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix} = r \begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix}$$

4 inconnues pour seulement 3 équations, comment va-t-on trouver b_0, b_1, b_2 et r ?

► **Réponse**

```

E = Eigen{Float64, Float64, Matrix{Float64}, Vector{Float64}}
values:
3-element Vector{Float64}:
 0.99999999999999978
 2.00000000000000007
 2.99999999999999925
vectors:
3x3 Matrix{Float64}:
 0.762001  -0.588348  0.534522
-0.635001   0.784465 -0.801784
 0.127     -0.196116  0.267261

```

```

1 E = eigen([
2     0 0 6
3     1 0 -11
4     0 1 6
5 ])

```

On trouve la première racine

```
► (1.0, [0.762001, -0.635001, 0.127])
```

```
1 r, b = E.values[1], E.vectors[:, 1]
```

En normalisant, on trouve bien $(x - 2)(x - 3)$

```
► [6.0, -5.0, 1.0]
```

```
1 b / b[3]
```

$6 - 5 \cdot x + x^2$

```
1 p2 * p3
```

Utilitaires ⇄

1 using Polynomials, LinearAlgebra, PlutoUI, Plots

```
> Precompiling Polynomials...
 1427.6 ms ✓ RecipesBase
 21192.0 ms ✓ Polynomials
 2 dependencies successfully precompiled in 23 seconds. 19 already precompiled.
Precompiling Plots...
 408.8 ms ✓ JLLWrappers
 800.2 ms ✓ Graphite2_jll
 937.0 ms ✓ OpenSSL_jll
 995.8 ms ✓ Libmount_jll
 673.3 ms ✓ EpollShim_jll
 827.6 ms ✓ Bzip2_jll
 911.7 ms ✓ LLVMOpenMP_jll
 848.1 ms ✓ Xorg_libICE_jll
 724.0 ms ✓ libpng_jll
 749.4 ms ✓ Xorg_libXau_jll
 769.1 ms ✓ libfdk_aac_jll
 811.5 ms ✓ LERC_jll
 862.8 ms ✓ LAME_jll
 821.8 ms ✓ fzf_jll
1069.1 ms ✓ JpegTurbo_jll
1227.6 ms ✓ XZ_jll
 761.2 ms ✓ Ogg_jll
5111.6 ms ✓ ColorSchemes
 916.3 ms ✓ mtdev_jll
 804.3 ms ✓ gperf_jll
 845.8 ms ✓ Xorg_libXdmcp_jll
1029.9 ms ✓ x265_jll
 806.0 ms ✓ x264_jll
```

1 import DocumenterCitations

```
> Precompiling DocumenterCitations...
 697.7 ms ✓ StringEncodings
 744.0 ms ✓ Git_jll
 427.4 ms ✓ Git
1616.2 ms ✓ YAML
10661.3 ms ✓ JSON3
 654.4 ms ✓ JSONSchema
 777.0 ms ✓ BibParser
1323.0 ms ✓ Bibliography
27757.9 ms ✓ Documenter
 2616.6 ms ✓ DocumenterCitations
10 dependencies successfully precompiled in 32 seconds. 58 already precompiled.
Precompiling FileIOExt...
 2509.0 ms ✓ Plots → FileIOExt
 1 dependency successfully precompiled in 3 seconds. 184 already precompiled.
```


qa (generic function with 2 methods)

```
1 include("utils.jl")
```

biblio =

►CitationBibliography("/home/runner/work/LSINC1113/LSINC1113/Lectures/biblio.bib", AlphaSt

```
1 biblio = load_biblio!()
```

① Loading bibliography from `/home/runner/work/LSINC1113/LSINC1113/Lectures/biblio.bib`...

⊗ Entry west2022Introduction is missing the publisher field(s).

① Loading completed.

cite (generic function with 1 method)

refs (generic function with 1 method)

```
1 refs(keys) = bibrefs(biblio, keys)
```

draw_angle (generic function with 1 method)

```
1 function draw_angle(r, from, to, label)
2     θs = range(from, stop=to, length = 10)
3     plot!(r * cos.(θs), r * sin.(θs), label = nothing, linestyle = :dash, color =
4           :black, arrow = Plots.arrow(:both, :closed))
5     _r = r + 0.02
6     θ_mid = (from + to) / 2
7     annotate!(_r * cos(θ_mid), _r * sin(θ_mid), text("\$" * label * "\$", :left))
8 end
```

draw_complex! (generic function with 1 method)

```
1 function draw_complex!(r, θ; cartesian = true, a_pos = :top, b_label = "b", θ_label  
  = "\\theta", start_θ = 0, θ_r = 1/6)  
2     a = r * cos(θ)  
3     b = r * sin(θ)  
4     plot!([0, a], [0, b], arrow = true, label = nothing, linewidth=2, color =  
       :black)  
5     if cartesian  
6         arrow = Plots.arrow(:both, :closed)  
7         plot!([a, a], [0, b], label = nothing, linestyle = :dash, color = :black,  
           arrow = arrow)  
8         plot!([0, a], [b, b], label = nothing, linestyle = :dash, color = :black,  
           arrow = arrow)  
9         annotate!(a/2, b, text("\\$a\\$", a_pos))  
10        annotate!(a - 0.02, b/2, text("\\$" * b_label * "\\$", :right))  
11        annotate!(a/2, b/2, text("\\$r\\$", :bottom, :right))  
12    end  
13    draw_angle(r * θ_r, 0, θ, θ_label)  
14 end
```

draw_complex (generic function with 1 method)

```
1 function draw_complex(args...; kws...)  
2     scatter([0], [0], ratio = :equal, color = :black, label = nothing, framestyle =  
       :origin, ticks = nothing)  
3     draw_complex!(args...; kws...)  
4 end
```