BOOK: Linear Algebra: Theory, Intuition, Code

AUTHOR: Mike X Cohen

WEBSITE: sincxpress.com

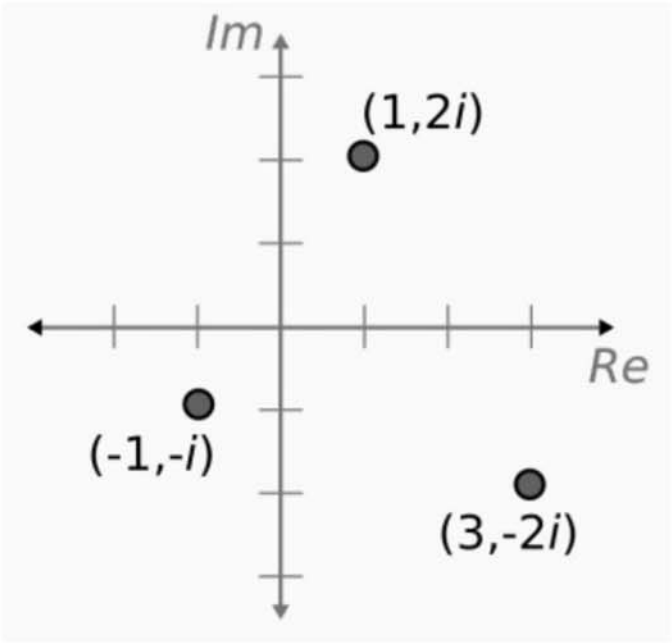
CHAPTER: Complex numbers (chapter 9)

In [1]:
```
## import libraries for the entire chapter
import numpy as np
```

In [ ]:
```

```

Section 9.1, COMPLEX NUMBERS AND "C"



Complex-Numbers

Section 9.2, COMPLEX VECTORS and MATRICES

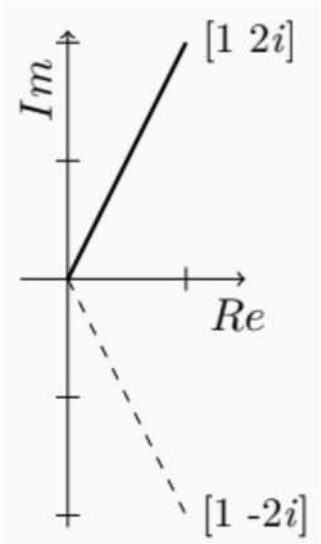
$$\begin{bmatrix} 6-i2 \\ 3+i4 \end{bmatrix}, \quad \begin{bmatrix} 1 \\ 2 \\ 3+i4 \\ i \end{bmatrix}, \quad \begin{bmatrix} 1+3i & 2-7i & -4+i \\ 4 & 7+2i & 4i \end{bmatrix}, \quad \begin{bmatrix} 7-4i & 1+i \\ 1+i & 2+2i \end{bmatrix}$$

In [4]:
```
# one way to create a complex number
z = complex(3,4)
```

```
# note the datatype input
Z = np.zeros(2,dtype=complex)
Z[0] = 3+4j
print(Z)
```

```
[3.+4.j 0.+0.j]
```

Section 9.3, COMPLEX CONJUGATE



$$z^* = \bar{z} = \overline{\mathrm{a}+\mathrm{b}i} = \mathrm{a}\text{-}\mathrm{b}i \tag{9.2}$$

$$\bar{z} = \overline{re^{i\theta}} = re^{-i\theta} \tag{9.3}$$

In [5]:
```
# real numbers for the matrix
r = np.random.randint(-3,4,size=3) # real part
i = np.random.randint(-3,4,size=3) # imag part

# create a complex matrix
Z = r + i*1j

# print the matrix and its transpose
print(Z)
print(Z.conj())
```

```
[ 0.-1.j  1.+2.j -3.-1.j]
[ 0.+1.j  1.-2.j -3.+1.j]
```

Section 9.4, ARITHMETIC with COMPLEX NUMBERS

ADDITION and SUBTRACTION

$$z + w = \mathrm{a} + i\mathrm{b} + \mathrm{c} + i\mathrm{d}$$

$$= (\mathrm{a}+\mathrm{c}) + i(\mathrm{b}+\mathrm{d})$$

MULTIPLICATION

$$
\begin{aligned}
zw &= (\mathrm{a}+i\mathrm{b})(\mathrm{c}+i\mathrm{d}) && (9.4)\\
&= \mathrm{ac}+i\mathrm{ad}+i\mathrm{bc}+i^2\mathrm{bd} && (9.5)\\
&= \mathrm{ac\text{-}bd}+i(\mathrm{ad}+\mathrm{bc}) && (9.6)
\end{aligned}
$$

DIVISION

$$
\begin{aligned}
\frac{z}{w} &= \frac{\mathrm{a}+i\mathrm{b}}{\mathrm{c}+i\mathrm{d}}\\
&= \frac{(\mathrm{c}-i\mathrm{d})(\mathrm{a}+i\mathrm{b})}{(\mathrm{c}-i\mathrm{d})(\mathrm{c}+i\mathrm{d})}\\
&= \frac{(\mathrm{c}-i\mathrm{d})(\mathrm{a}+i\mathrm{b})}{\mathrm{c}^2+\mathrm{d}^2}\\
&= \frac{(\mathrm{ca}+\mathrm{db})+i(\mathrm{cb\text{-}da})}{\mathrm{c}^2+\mathrm{d}^2}
\end{aligned}
$$

Section 9.5, The HERMITIAN and COMPLEX DOT PRODUCTS

$$
\begin{bmatrix}2 & 4-i5 & 1 & 2+i9\end{bmatrix}^{\mathrm{H}} = \begin{bmatrix}2\\4+i5\\1\\2-i9\end{bmatrix}
$$

; Hermitian

$$
\mathbf{v} = \begin{bmatrix}0\\i\end{bmatrix}. \quad \begin{cases}\mathbf{v}^{\mathrm{T}}\mathbf{v} = 0^2+i^2 & = -1\\ \mathbf{v}^{\mathrm{H}}\bar{\mathbf{v}} = 0^2+(-i)(-i) & = -1\\ \mathbf{v}^{\mathrm{H}}\mathbf{v} = 0^2+(-i)(i) & = 1\\ \mathbf{v}^{\mathrm{T}}\bar{\mathbf{v}} = 0^2+(i)(-i) & = 1\end{cases}
$$

; dot-product

Proprietà

Ogni matrice hermitiana è una matrice quadrata della forma $A = B + iC$, dove B è una matrice simmetrica (uguale alla propria trasposta) a componenti reali e C è una matrice antisimmetrica (opposta alla propria trasposta) a componenti reali, e viceversa. In particolare, gli elementi sulla diagonale principale di una matrice hermitiana sono reali, ed una matrice a componenti reali è hermitiana se e solo se è simmetrica.

Sono matrici hermitiane la somma di due matrici hermitiane e l'inversa di una matrice hermitiana invertibile. Il prodotto di due matrici hermitiane A e B, invece, è una matrice hermitiana se e solo se queste commutano, cioè se $AB = BA$.

In [6]:
```
v = [0,1j]

# "normal" and hermitian dot product
print(np.dot(v,v))
print(np.vdot(v,v))
```

```
(-1+0j)
(1+0j)
```

Section 9.6, SPECIAL COMPLEX MATRICES

Una matrice Hermitiana è l'equivalente con valori complessi di qualcosa tra una matrice simmetrica ($\mathbf{A} = \mathbf{A}^T$) e una matrice antisimmetrica ($\mathbf{A} = -\mathbf{A}^T$). Una matrice Hermitiana è definita come $\mathbf{A} = \mathbf{A}^H$. Pertanto, le grandezze della parte reale e della parte immaginaria sono le stesse, ma i segni delle parti immaginarie sono invertiti.

$$\begin{bmatrix} 2 & 3-2i & 2+2i \\ 3+2i & 5 & 8 \\ 2-2i & 8 & 9 \end{bmatrix} \tag{9.9}$$

Hermitian

$$\frac{1}{2}\begin{bmatrix} 1+i & 1-i \\ 1-i & 1+i \end{bmatrix}$$

Unitary

Section 9.10, code block 9.7

In [8]:
```
U = .5*np.array([ [1+1j,1-1j],[1-1j,1+1j] ])

# Hermitian
print(U@np.matrix(U).H,'  U @ U.H = unit matrix --> U is Hermitian')
print(' ')
# not Hermitian
print(U@U.T,'  U @ U.T --> not Hermitian')
```

```
[[1.+0.j 0.+0.j]
 [0.+0.j 1.+0.j]]   U @ U.H = unit matrix --> U is Hermitian

[[0.+0.j 1.+0.j]
 [1.+0.j 0.+0.j]]   U @ U.T --> not Hermitian
```

Section 9.10, code block 9.9

In [10]:
```
# create a complex matrix
r = np.random.randn(3,3)
i = np.random.randn(3,3)
A = np.matrix( r + i*1j )

# new matrices by adding and multiplying
A1 = A+A.H
A2 = A@A.H

# test for Hermitian
print(A1-A1.H)
print(' ')
print(A2-A2.H,'  computeristicamente = matrice nulla')
```

```
[[0.+0.j 0.+0.j 0.+0.j]
 [0.+0.j 0.+0.j 0.+0.j]
 [0.+0.j 0.+0.j 0.+0.j]]

[[0.-1.11301671e-16j 0.+0.00000000e+00j 0.+0.00000000e+00j]
 [0.+0.00000000e+00j 0.+5.06402426e-16j 0.+0.00000000e+00j]
 [0.+0.00000000e+00j 0.+0.00000000e+00j 0.-1.92225358e-17j]]   computeristicament
e = matrice nulla
```

In [ ]: