

LAbook_chapter02

July 10, 2024

1 BOOK: Linear Algebra: Theory, Intuition, Code

AUTHOR: Mike X Cohen

WEBSITE: sincxpress.com

1.1 CHAPTER: Vectors (chapter 2)

1.1.1 Hanno un valore, una direzione, non trattano il punto di applicazione. Le coordinate – invece – fanno riferimento ad uno specifico punto dello spazio

```
[6]: ## import libraries for the entire chapter  
import numpy as np  
import matplotlib.pyplot as plt
```

1.1.2 Section 2.1, code block 2.1 - Scalars

```
[2]: # creating scalars (numeric variables)  
aScalar = 4
```

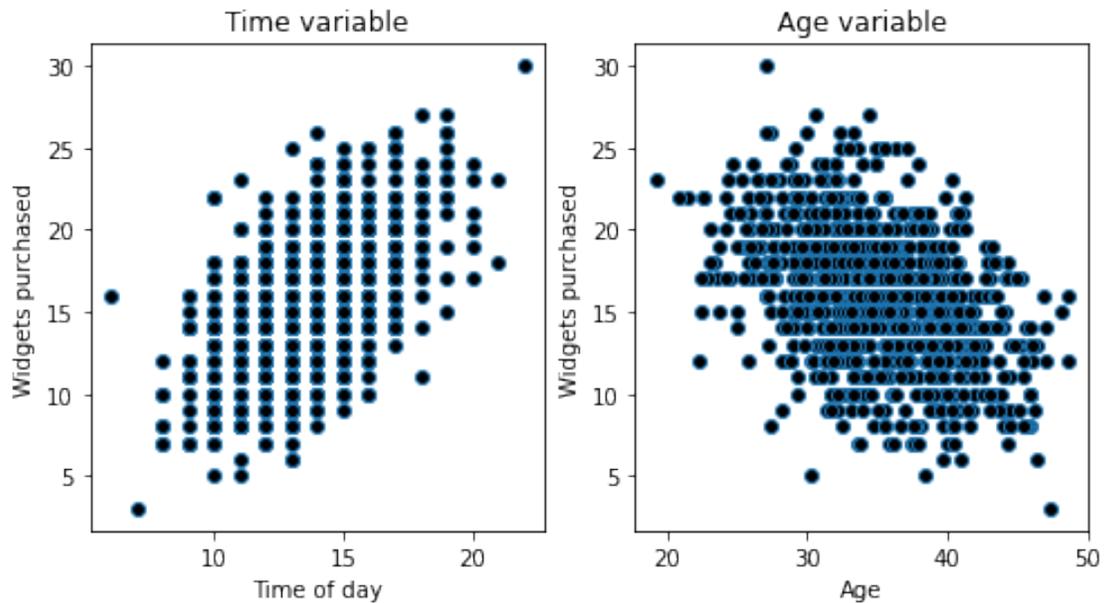
1.1.3 Section 2.2, code block 2.3 - Vectors: geometry and algebra

1.1.4 Un vettore è una lista ordinata di numeri. Il loro numero è detto “dimensione del vettore”.

1.1.5 Per i vettori useremo le parentesi quadre. Matematicamente vettori a molte dimensioni sono facili da scrivere, ma come visualizzazione ci dobbiamo fermare a 2D, 3D. Le componenti di un vettore possono essere funzioni.

```
[25]: # make the vector  
import sympy as sym  
t = np.linspace(0., np.pi, 10)  
v1=np.array([np.cos(t), np.sin(t), t])  
v = np.array([2, -1])  
  
# plot it  
plt.plot([0, v[0]], [0, v[1]])  
plt.plot(t, v1[0]+1)  
plt.plot(t, v1[1]+1)
```

```
plt.axis('square')
plt.axis([-1,3,-1,3])
plt.scatter([0.],[0.])
plt.grid('on')
plt.show()
```



1.1.6 Section 2.2, code block 2.5

```
[15]: # list
v1 = [2,5,4,7]

# array, no orientation
v2 = np.array([2,5,4,7])

# column vector
v3 = np.array([ [2],[5],[4],[7] ])
print(v3," = Vettore colonna")
print(" ")

# row vector
v4 = np.array([ [2,5,4,7] ])
print(v4," = Vettore riga")
```

```
[[2]
 [5]
 [4]
 [7]] = Vettore colonna
```

[[2 5 4 7]] = Vettore riga

1.1.7 Section 2.3, code block 2.7 - Transpose operation

1.1.8 Operazione di trasposizione di un vettore

```
[28]: # row vector
v1 = np.array([ [2,5,4,7] ])

# column vector
v2 = v1.T
print(v1, " = matrice v1")
print("")
print(v2, " = v2 = trasposta di v1")
v3 = v2.T
print("")
print(v3, " = v3 = trasposta di v2 = v1")
```

[[2 5 4 7]] = matrice v1

```
[[2]
 [5]
 [4]
 [7]] = v2 = trasposta di v1
```

[[2 5 4 7]] = v3 = trasposta di v2 = v1

1.1.9 Section 2.4, code block 2.7.1 - Vector addition and subtraction

1.1.10 Somma e sottrazione di vettori

```
[44]: # two vectors
v1 = np.array([ [2,5,4,7] ])
v2 = np.array([ [3,1,0,2] ])
v4 = np.array([ [[2],[5],[4],[7]] ])
v5 = np.array([ [[3],[1],[0],[2]] ])
v3 = v1+v2
print(v3, " = v3 somma dei vettori riga v1 e v2")
print("")
v6 = v4+v5
print(v6, " = v3 somma dei vettori colonna v4 e v5")
```

[[5 6 4 9]] = v3 somma dei vettori riga v1 e v2

```
[[[5]
 [6]
 [4]
 [9]]] = v3 somma dei vettori colonna v4 e v5
```

1.1.11 Section 2.5, code block 2.9 - Vector-scalar multiplication

1.1.12 Prodotto scalare

```
[31]: # two vectors
v1 = np.array([2,5,4,7])
v2 = np.array([4,1,0,2])

# scalar-multiply and add
v3 = 4*v1
print(v3," = prodotto scalare tra l'intero 4 e v1")
```

[8 20 16 28] = prodotto scalare tra l'intero 4 e v1

1.1.13 Section 2.6, Exercises

1.1.14 Section 2.7, Answers

1.1.15 Section 2.8, Code challenges

1.1.16 Section 2.9, Code solutions

```
[ ]:
```