

numpy

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

1 Opérations terme à terme

```
V1 = np.array([1,2,3])  
V2 = np.array([3,4,5])  
V1 + V2
```

```
array([4, 6, 8])
```

```
# par contre les éléments d'un vecteur numpy doivent être  
# tous du même type (ici des entiers)  
V1.dtype
```

```
dtype('int64')
```

```
W = V1**2 + V2 / 2  
W
```

```
array([ 2.5,  6. , 11.5])
```

```
W.dtype
```

```
dtype('float64')
```

```
np.array(W, dtype="int64") # on force le type
```

```
array([ 2,  6, 11])
```

```
V1 > 2 # encore une opération terme à terme
```

```
array([False, False,  True], dtype=bool)
```

```
V = np.array([0, np.pi/6, np.pi/2, np.pi])  
  
# les fonctions mathématiques de la bibliothèque numpy  
# agissent aussi terme à terme  
np.sin(V)
```

```
array([ 0.00000000e+00,  5.00000000e-01,  1.00000000e+00,  
       1.22464680e-16])
```

1.1 Application au tracé de courbes

```
X = np.linspace(0, 2*np.pi, 250) # 250 points équirépartis  
plt.plot(X, np.sin(X), X, np.cos(X))
```

```
[<matplotlib.lines.Line2D at 0x7fa817800e80>,  
<matplotlib.lines.Line2D at 0x7fa817808080>]
```

2 Indices et accès aux éléments

```
A = np.array([[1,2,3], [-3,-2,-1], [1,0,1]])  
A
```

```
array([[ 1,  2,  3],  
       [-3, -2, -1],  
       [ 1,  0,  1]])
```

```
A[0][1]
```

```
2
```

```
A[0, 1] # notation spéciale numpy
```

```
2
```

```
A[[0,2]] # on peut même passer la liste des indices  
# que l'on veut récupérer
```

```
array([[1, 2, 3],  
       [1, 0, 1]])
```

```
A[0, 0:2] # comme pour les listes, on peut utiliser des slices
```

```
array([1, 2])
```

```
A[0:2, 0:2] # et même sur chaque indice
```

```
array([[ 1,  2],  
       [-3, -2]])
```

2.1 Comportement de l'affectation

```
I3 = np.identity(3) # matrice identité  
I3
```

```
array([[ 1.,  0.,  0.],  
       [ 0.,  1.,  0.],  
       [ 0.,  0.,  1.]])
```

```
L = [[int(i==j) for j in range(3)] for i in range(3)]  
L
```

```
[[1, 0, 0], [0, 1, 0], [0, 0, 1]]
```

```
t = I3[0] # ceci est seulement une "vue"  
# et non le contenu de la première ligne  
I3[0] = I3[1]  
I3[1] = t  
# d'où ce résultat  
I3
```

```
array([[ 0.,  1.,  0.],  
       [ 0.,  1.,  0.],  
       [ 0.,  0.,  1.]])
```

```
t = L[0]  
L[0] = L[1]  
L[1] = t  
L
```

```
[[0, 1, 0], [1, 0, 0], [0, 0, 1]]
```