

Arithmétique avec Numpy

Arithmétique
Données aléatoires
Fonctions de base

Programmation Python Partie 2 : Arithmétique avec NumPy, nombres aléatoires et fonctions de base

Arithmétique avec Numpy

Arithmétique
Données aléatoires
Fonctions de base

Arithmétique sur des tableaux NumPy

```
In [40]: a = np.arange(5, dtype=float)
         print(a)
```

```
[ 0.  1.  2.  3.  4.]
```

```
In [41]: print(a + 1)
```

```
[ 1.  2.  3.  4.  5.]
```

```
In [42]: print(2 * a)
```

```
[ 0.  2.  4.  6.  8.]
```

```
In [43]: print(a ** 2)
```

```
[ 0.  1.  4.  9. 16.]
```

Arithmétique avec Numpy

Arithmétique
Données aléatoires
Fonctions de base

Arithmétique sur des tableaux NumPy (suite)

```
In [44]: a = np.arange(5, dtype=float)
         print(a)

[ 0.  1.  2.  3.  4.]
```

```
In [45]: print(a + a) # addition élément par élément

[ 0.  2.  4.  6.  8.]
```

```
In [46]: print(a * a) # multiplication élément par élément

[ 0.  1.  4.  9. 16.]
```

Arithmétique avec Numpy

Arithmétique

Données aléatoires

Fonctions de base

Le problème de la division entière

```
In [47]: a = np.array([1, 2], dtype=np.int64)
         b = np.array([2, 2], dtype=np.int64)
         print(a / b)
```

```
[0 1]
```

```
In [48]: a = np.array([1, 2], dtype=float)
         b = np.array([2, 2], dtype=float)
         print(a / b)
         print(a // b)
```

```
[ 0.5  1. ]
```

```
[ 0.  1.]
```

Arithmétique avec Numpy

Arithmétique
Données aléatoires
Fonctions de base

Données aléatoires

```
In [49]: from numpy import random
```

```
In [50]: # tirage uniforme dans [0,1]
random.random_sample((3, 3)) # ou random.rand
```

```
Out[50]: array([[ 0.39901777,  0.6412892 ,  0.97677018],
                [ 0.88608808,  0.2866462 ,  0.19678252],
                [ 0.84092386,  0.39505946,  0.78899804]])
```

```
In [51]: # tirage suivant une loi normale standard
random.standard_normal((3, 3)) # ou random.randn
```

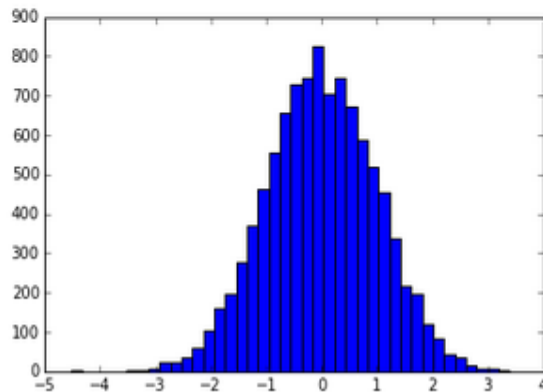
```
Out[51]: array([[ 0.08910219, -0.76917745, -1.59923397],
                [-0.31889444, -0.04475295, -1.17248134],
                [ 0.88410981, -1.10103646,  1.45595075]])
```

Arithmétique avec Numpy

Arithmétique
Données aléatoires
Fonctions de base

Affichage de l'histogramme des tirages

```
In [52]: a = random.standard_normal(10000)  
         hh = plt.hist(a, 40)
```



Arithmétique avec Numpy

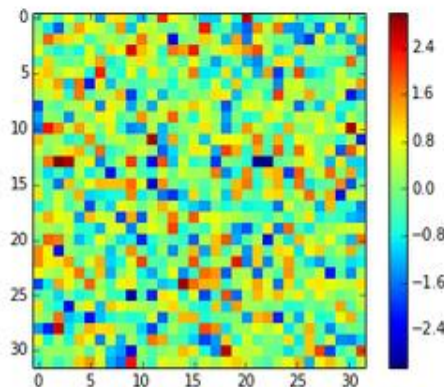
Arithmétique
Données aléatoires
Fonctions de base

Affichage des tirages comme une image

```
In [53]: C = random.standard_normal((32, 32))  
         print(C.shape)
```

```
(32, 32)
```

```
In [54]: plt.imshow(C, interpolation='nearest')  
         plt.colorbar()  
         plt.show()
```



Arithmétique avec Numpy

Arithmétique
Données aléatoires
Fonctions de base

zeros et ones

```
In [55]: # Création d'un tableau 2D de zeros
A = np.zeros((2, 3)) # Attention: zeros(2, 3) est FAUX
print(A)
print(A.dtype)
```

```
[[ 0.  0.  0.]
 [ 0.  0.  0.]]
float64
```

```
In [56]: print(np.zeros((1, 3), dtype=int)) # ligne
```

```
[[0 0 0]]
```

```
In [57]: print(np.zeros((3, 1), dtype=int)) # colonne
```

```
[[0]
 [0]
 [0]]
```

```
In [58]: print(np.ones((3, 3))) # tableau 3x3 de 1
```

```
[[ 1.  1.  1.]
 [ 1.  1.  1.]
 [ 1.  1.  1.]]
```


Arithmétique avec Numpy

Arithmétique
Données aléatoires
Fonctions de base

diag

```
In [59]: # création d'une matrice diagonale  
D = np.diag([1, 2, 3])  
print(D)
```

```
[[1 0 0]  
 [0 2 0]  
 [0 0 3]]
```

```
In [60]: # extraction de la diagonale  
print(np.diag(D))
```

```
[1 2 3]
```