

Les bases de SciPy

La librairie SciPy ajoute à NumPy des fonctionnalités mathématiques.

Directive d'importation

- Méthode standard :

```
import scipy as sp
```

- Importation par sous-modules (cf le [site de SciPy](#)) :

```
from scipy import optimize
from scipy import interpolate
from scipy import integrate
...
```

Fonctionnalités

La librairie SciPy est particulièrement intéressante pour ces méthodes numériques :

- intégrales numériques
- intégration d'équations différentielles ordinaires
- Recherche de racines d'équations
- minimisation de fonctions
- modélisation par moindres carrés
- fonctions spéciales
- transformées de Fourier
- analyse du signal
- interpolation
- algèbre linéaire, y compris les problèmes aux valeurs propres et vecteurs propres

Des informations générales sur ces techniques numériques peuvent être trouvées notamment sur le site de [Numerical Recipes](#). En particulier, les anciennes éditions sont [accessibles gratuitement](#) à la lecture. Les méthodes et algorithmes sont transposables à des langages comme le Python, y compris avec l'utilisation de bibliothèques comme SciPy.

Intégrales définies

La librairie `scipy.integrate` propose plusieurs programmes permettant de calculer des intégrales définies et intégrer des [équations différentielles ordinaires](#) (ODE)

Exemple de calcul avec quad :

[integrate-01.py](#)

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
"""
```

```
Intégration numérique. Références :
http://docs.scipy.org/doc/scipy/reference/integrate.html
http://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.quad.html
quad retourne un tuple dont le premier élément est l'intégrale définie
calculée
"""
from scipy.integrate import quad

def f(x):
    return x**4.

I = quad(f,0,2)
print(I, I[0])
```

Constantes physiques

[scipy.constants-01.py](#)

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
https://docs.scipy.org/doc/scipy-1.0.0/reference/constants.html
"""
import scipy.constants
for key, val in scipy.constants.physical_constants.items():
    print(key, val)
    print(key, scipy.constants.value(key), scipy.constants.unit(key),
scipy.constants.precision(key))

print(scipy.constants.find('Boltzmann'))
print(scipy.constants.Boltzmann)
print(scipy.constants.physical_constants['Boltzmann constant'])
print(scipy.constants.value('Boltzmann constant'))
print(scipy.constants.unit('Boltzmann constant'))
print(scipy.constants.precision('Boltzmann constant'))
```

Références

- [SciPy tutorial](#)
- [Reference guide](#)
- [Scipy : high-level scientific computing](#), de Adrien Chauve, Andre Espaze, Emmanuelle Guillard, Gaël Varoquaux, Ralf Gommers
- [Cookbook](#)
 - [Lotka-Volterra](#)
 - [Mouvement brownien](#)
- <https://uiuc-cse.github.io/2014-01-30-cse/lessons/thw-scipy/tutorial.html>

From:

<https://dvillers.umons.ac.be/wiki/> - **Didier Villers, UMONS - wiki**

Permanent link:

https://dvillers.umons.ac.be/wiki/teaching:progappchim:scipy_simple

Last update: **2019/03/22 12:07**

