Les bases de Matplotlib, une librairie pour réaliser des graphiques 2D

Matplotlib est une bibliothèque très puissante du langage de programmation Python destinée à tracer et visualiser des données sous formes de graphiques. Elle est souvent combinée avec les bibliothèques python de calcul scientifique :

- NumPy : gestion de tableaux numériques multidimensionnels, algèbre linéaire, transformées de Fourier, nombres (pseudo-)aléatoires
- SciPy : méthodes numériques comme l'intégration ou l'optimisation
- SymPy : mathématiques symboliques
- Pandas: analyse de données

Avec Matplotlib, on peut créer rapidement un graphe à partir de deux listes (voir le premier exemple ci-après).

Matplotlib permet de générer facilement des graphiques, camemberts ou autres histogrammes, intégrant symboles, barres d'erreur, éléments colorés,... Il peut créer pratiquement tous les types connus de graphiques (consulter la galerie d'exemples).

Le projet Pylab vise à regrouper ces différentes librairies. De nombreuses commandes de Pylab ont été définies semblablement aux commandes du logiciel commercial MatLab.

Installation

La page d'installation de Matplotlib fournit une procédure pas à pas assez complète et facile pour installer matplotlib (et NumPy). Sinon :

- Sous Windows, installez une distribution complète comme Python (x, y)
- Sous Linux, installez les librairies suivantes : python-numpy python-scipy python-matplotlib

Directive d'importation

- standard :

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

- alternative, simplifiée (en mode pylab) :

```
from pylab import *
```

Graphiques de séries de points en lignes

Il s'agit d'un graphe classique de séries de points reliés par une ligne colorée : <sxh python; title : simple_series_01.py> #! /usr/bin/env python # -*- coding: utf-8 -*- “”“ Matplotib : graphe simple de séries de données ”“”

import matplotlib.pyplot as plt #directive d'importation standard

plt.figure() #initialisation d'une nouvelle figure

#les données serie_x = [0.,1.,2.,3.,5.,7.,11.,13.,17.,19.] serie_y1 = [x**2 for x in serie_x] serie_y2 = [x**1.5 for x in serie_x]

#plot de deux lignes lignes plt.plot(serie_x, serie_y1) plt.plot(serie_x, serie_y2)

plt.savefig(“example.png”) # sauvegarde de la figure

plt.show() # vue interactive de la figure </sxh>

Le même graphique peut être agrémenté d'un titre, d'appellations pour les axes, de valeurs limites, d'une légende :

<sxh python; title : simple_series_02.py> #! /usr/bin/env python # -*- coding: utf-8 -*- “”“ Matplotib : graphe simple de séries de données ajouts des titres pour la figure, les axes, d'une légende, de valeurs min et max ”“”

import matplotlib.pyplot as plt #directive d'importation standard

plt.figure() #initialisation d'une nouvelle figure plt.title(u“Ma première figure avec Matplolib”) # la chaine unicode est préfixée par u (présence de caractères non ASCII primitifs)

#les données serie_x = [0.,1.,2.,3.,5.,7.,11.,13.,17.,19.] serie_y1 = [x**2 for x in serie_x] serie_y2 = [x**1.5 for x in serie_x]

plt.xlim(0, 20.) #les limites suivant x plt.ylim(-5, 400.) # les limites suivant y

#plot de deux lignes lignes plt.plot(serie_x, serie_y1, label=“x^2”) plt.plot(serie_x, serie_y2, label=“x^1.5”) plt.xlabel(u“Les données X”) plt.ylabel(u“Des valeurs calculées de Y”)

#ajout d'une légende plt.legend()

plt.savefig(“example.pdf”) # sauvegarde de la figure au format pdf

plt.show() # vue interactive de la figure </sxh>

Changer simplement les lignes en points

La personnalisation des lignes ou point se fait très facilement lorsqu'on utilise la fonction de traçage, en passant dans une chaîne de deux caractères la spécification du type de ligne et de la couleur à

utiliser. Voici un exemple d'un graphique qui utilise des cercles verts comme marqueurs, sans ligne, grâce aux deux caractères “g” (pour green ou vert) et “o” (pour figurer des petits cercles. <sxh python; title : simple_series_03.py> #! /usr/bin/env python # -*- coding: utf-8 -*- “”“ Matplotib : graphe simple de séries de données utilisation de points ”“”

import matplotlib.pyplot as plt #directive d'importation standard

plt.figure() #initialisation d'une nouvelle figure

#les données serie_x = [0.,1.,2.,3.,5.,7.,11.,13.,17.,19.] serie_y1 = [x**2 for x in serie_x]

#plot plt.plot(serie_x, serie_y1, “go”)

plt.show() # vue interactive de la figure

</sxh> Pour essayer d'autres marqueurs ou couleurs, consultez la documentation ici :

- http://matplotlib.org/api/markers_api.html
- http://matplotlib.org/api/colors_api.html?highlight=colors#module-matplotlib.colors
- http://matplotlib.org/users/pyplot_tutorial.html#controlling-line-properties

Tracé d'une fonction

<sxh python; title : simple_fonction_01.py> #! /usr/bin/env python # -*- coding: utf-8 -*- “”“ Matplotib : graphe simple d'une fonction : cosinusoïde amortie

”“”

from pylab import * # directive d'importation simplifiée

def my_func(t):

```
  s1 = cos(2*pi*t*1.)  #essayez 2., 4., 100. !!
  e1 = exp(-t)
  return s1*e1
```

tvals = arange(0., 5., 0.05) # arange (importé) permet de définir un tableau numérique # arange([start], stop[, step]) renvoie un “array” de valeurs # espacées de step à partir de start jusque stop # passer step à 0.001 ?

plot(tvals, my_func(tvals), 'bo', tvals, my_func(tvals), 'k')

show() </sxh>

Histogramme simple

<sxh python; title : simple-histogram-random_numbers.py> #! /usr/bin/env python # -*- coding: utf-8 -*- “”“ Matplotib : histogramme simple de nombres aléatoires ”“” from pylab import randn, hist, show #importation simplifiée

x = randn(10000) hist(x, 25) show() </sxh>

Références

- Le site officiel
- Galerie d'exemples avec leur code
- Le guide utilisateur officiel
- Manuel en pdf (1311 pages)
- Cookbook Matplotlib
- Matplolib sur Wikipédia
- Une présentation synthétique en français
- Tutoriel en français
- Un tutoriel en anglais
- Matplotlib: plotting, par Nicolas Rougier, Mike Müller, Gaël Varoquaux (et la version dérivée de Nicolas Rougier)
- http://www.thetechrepo.com/main-articles/465-how-to-create-a-graph-in-python.html
- Un article intéressant sur les recommandations pour de bonnes figures : Ten Simple Rules for Better Figures, Nicolas P. Rougier (INRIA, France). Les figures sont crées avec matplotlib et l'ensemble de l'article est disponible sous licence CC0.