Тема 3. Введення Matplotlib

Вступ

matplotlib - Набір додаткових модулів (бібліотек) мови Python. Надає засоби для побудови найрізноманітніших 2D графіків та діаграм даних. Переваги цієї бібліотеки – простота використання. Для побудови дуже складних і барвисто оформлених діаграм достатньо кількох рядків коду. При цьому якість зображень більш ніж достатньо для їх опублікування. Робота цього модуля зазвичай має на увазі також використання модуля NumPy.

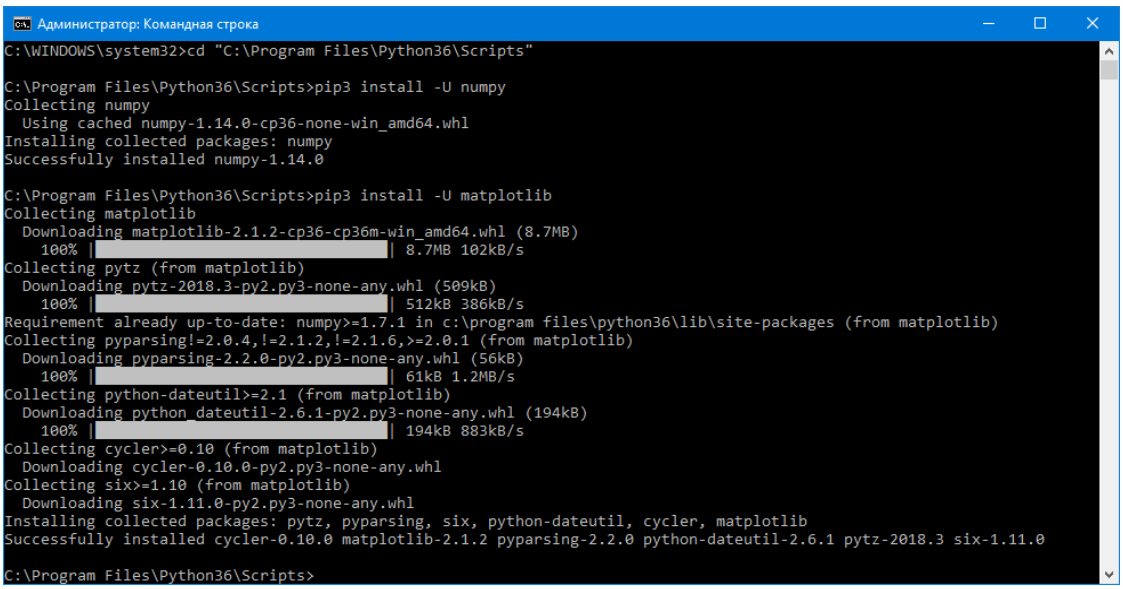
Мережевий ресурсhttps://github.com/whitehorn/Scientific_graphics_in_python містить загальнодоступний підручник. Багато інформації з цієї теми розміщено наhttp://jenyay.net/Programming/Python3dihttps://pyprog.pro/.

Встановлення

У Windows установка і matplotlib і NumPy не повинні викликати жодних проблем. Використовуємо pip3 - систему керування пакетами, яка використовується для встановлення та керування програмними пакетами, написаними на Python.

Якщо Python, наприклад, встановлений у C:\Program Files\Python36\, то в командному рядку адміністратора (це важливо) необхідно виконати такі команди:

cd ":\Program Files\Python36\Scripts"
pip3 install numpy
pip3 install matplotlib
чи так
python -m pip install numpy
python -m pip install matplotlib



Протокол роботи:

```
C:\WINDOWS\system32>cd "C:\Program Files\Python36\Scripts"

C:\Program Files\Python36\Scripts>pip3 install -U numpy
Collecting numpy
  Using cached numpy-1.14.0-cp36-none-win_amd64.whl
Installing collected packages: numpy
Successfully installed numpy-1.14.0

C:\Program Files\Python36\Scripts>pip3 install -U matplotlib
Collecting matplotlib
  Downloading matplotlib-2.1.2-cp36-cp36m-win_amd64.whl (8.7MB)
    100% |████████████████████████████████| 8.7MB 102kB/s
Collecting pytz (від matplotlib)
  Downloading pytz-2018.3-py2.py3-none-any.whl (509kB)
    100% |████████████████████████████████| 512kB 386kB/s
Requirement already up-to-date: numpy>=1.7.1 in c:\program
files\python36\lib\site-packages (from matplotlib)
Collecting pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 (from matplotlib)
  Downloading pyparsing-2.2.0-py2.py3-none-any.whl (56kB)
    100% |████████████████████████████████| 61kB 1.2MB/s
Collecting python-dateutil>=2.1 (from matplotlib)
  Downloading python_dateutil-2.6.1-py2.py3-none-any.whl (194kB)
    100% |████████████████████████████████| 194kB 883kB/s
Collecting cycler>=0.10 (від matplotlib)
  Downloading cycler-0.10.0-py2.py3-none-any.whl
Collecting six>=1.10 (from matplotlib)
  Downloading six-1.11.0-py2.py3-none-any.whl
Installing collected packages: pytz, pyparsing, six, python-dateutil,
cycler, matplotlib
Successfully installed cycler-0.10.0 matplotlib-2.1.2 pyparsing-2.2.0
python-dateutil-2.6.1 pytz-2018.3 six-1.11.0

C:\Program Files\Python36\Scripts>
```

Примітка-У старших версіях MatPlotLib можливі проблеми при інсталяції. У цьому випадку слід встановлювати Python та необхідні модулі для одного користувача (вказується в діалозі встановлення Python).

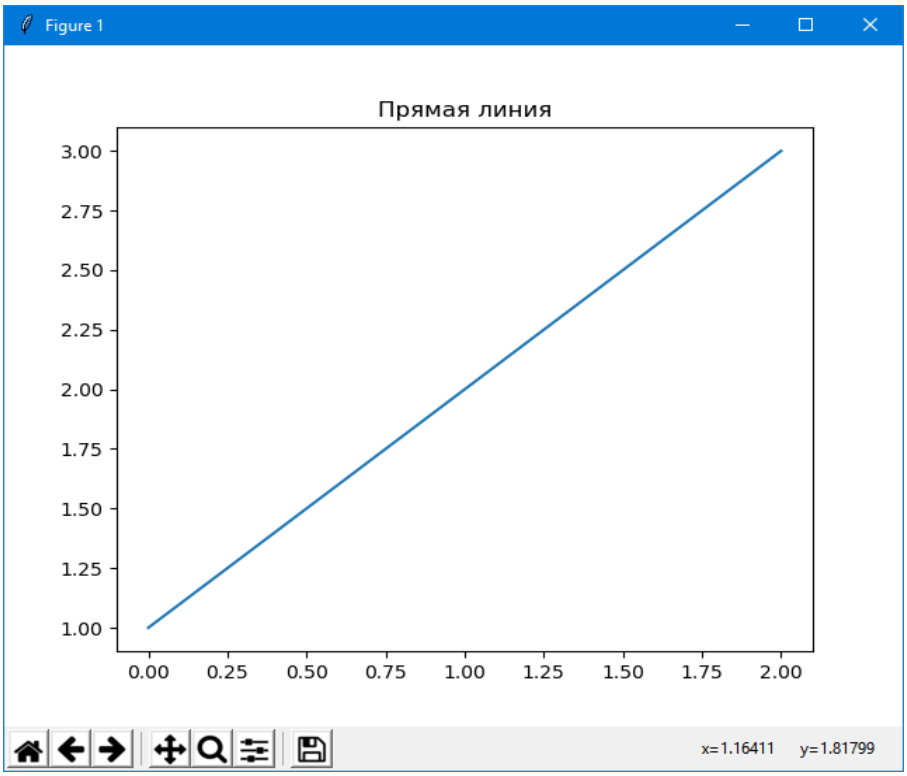
Перевірка

Після встановлення перевіряємо працездатність. Запускаємо консоль Python, вводимо:

```
>>> import matplotlib as mpl
>>> print ('Current version on matplotlib library is', mpl.__version__)
Current version on matplotlib library is 2.1.2
```

```
# де-факто стандарт виклику pyplot у python
import matplotlib.pyplot as plt
plt.plot([1,2,3])
[<matplotlib.lines.Line2D object at 0x0301E810>]
plt.title('Пряма лінія')
<matplotlib.text.Text object at 0x03062430>
plt.show()
```

В результаті має з'явитися таке вікно діаграми:



Що було зроблено?

З пакету matplotlib імпортований модуль pyplot під назвою plt.

Модуль pyplot містить функції (схожі на команди) створення діаграм та зміни властивостей їх елементів.

Функція plot() будує прямокутні двовимірні діаграми (графіки) координатах X - Y.

Якщо функції plot передано один аргумент (у нашому випадку - список [1,2,3]), вона розглядає його як сукупність значень відкладаються по осі Y, тоді осі X йому буде відповідати автоматично згенерований набір чисел 0,1,2 ... N - 1, де N - число елементів в переданому списку.

Функція title() визначає заголовок діаграми, а функція show() виводить інтерактивне вікно діаграми. Загалом, все дуже просто

Налаштування

Matplotlib можна легко налаштувати через конфігураційний файл matplotlibrc. Якщо Python встановлений в папку C:\Program Files\Python36\, то настроювальний файл розташовується в C:\Program Files\Python36\Lib\site-packages\matplotlib\mpl-data\.

Вносити зміни можна прямо тут, але краще скопіювати файл до папки користувача: C:\Documents and Settings\UserName\.matplotlib\, інакше при перевстановленні пакета конфігураційний файл буде перезаписано. Параметри пакета задаються у файлі як пар властивість : значення, символ # відокремлює коментар.

Властивість font.family визначає тип активного шрифту за умовчанням, може набувати п'ять значень: serif, sans-serif, cursive, fantasy, monospace.

У свою чергу властивості font.serif, font.sans-serif, font.cursive, font.fantasy, font.monospace містять списки імен шрифтів (через кому, в порядку зменшення пріоритету), що відповідають кожному типу.

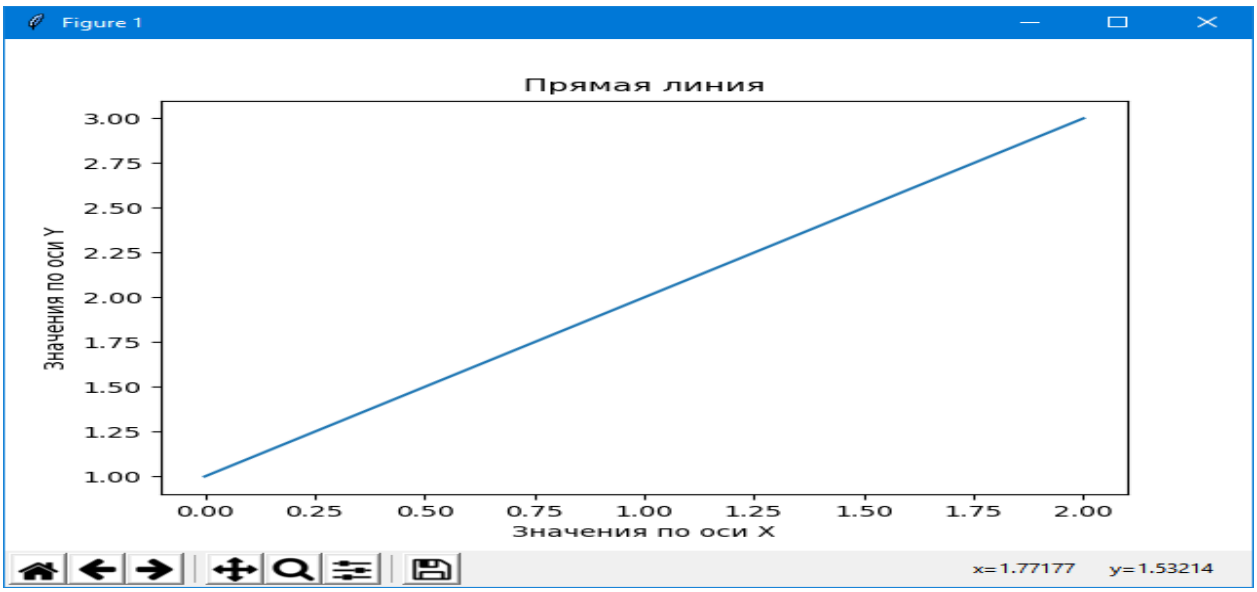
Наприклад, для "русифікації" matplotlib, можливо, доведеться змінити імена шрифтів у списку, на імена доступних в системі шрифтів, що містять кириличні символи. Наприклад так:

```
font.serif : Verdana, Arial
font.sans-serif : Tahoma, Arial
font.cursive: Courier New, Arial
font.fantasy: Comic Sans MS, Arial
font.monospace : Arial
```

Зауважимо, що останні версії пакета встановлюються з коректним настроювальним файлом.

Розглянемо приклад:

```
>>> plt.plot([1,2,3])
[<matplotlib.lines.Line2D object at 0x000001BB6DFCA588>]
>>> plt.title('Пряма лінія')
Text(0.5,1,'Пряма лінія')
>>> plt.xlabel(u'Значення по осі X')
Text(0.5,0,'Значення по осі X')
>>> plt.ylabel(u'Значення по осі Y')
Text(0,0.5,'Значення по осі Y')
>>> plt.show()
```



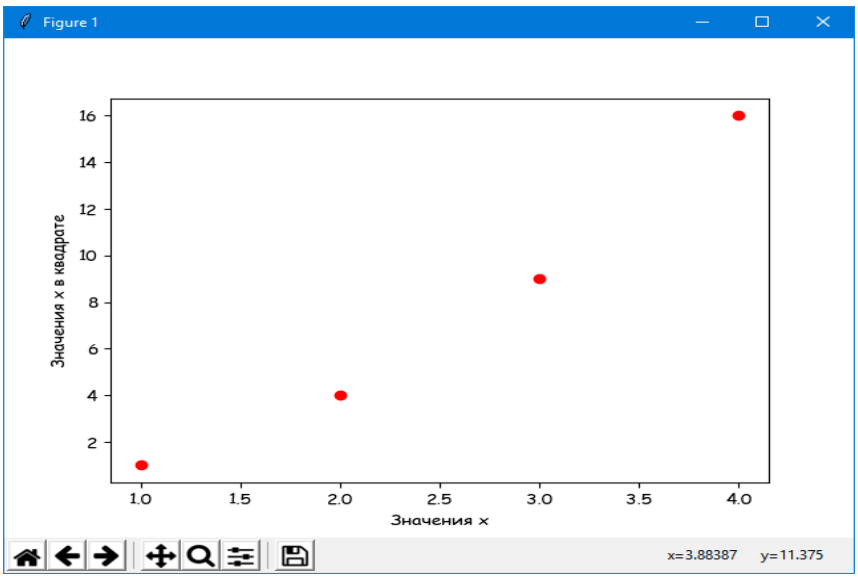
Функції pyplot.xlabel(), pyplot.ylabel(), як легко здогадатися за назвою, встановлюють підписи до осей X та Y відповідно.

Пакет matplotlib можна налаштовувати динамічно (під час виконання програми). У пакеті визначено структуру matplotlib.rcParams, з якою працюють зі словником. Ключові слова в даному випадку - імена властивостей файлу matplotlibrc.

Налаштування шрифтів через matplotlib.rcParams можна змінити, наприклад:

```
>>>import matplotlib as mpl
>>>import matplotlib.pyplot as plt
>>> mpl.rcParams['font.family'] = 'fantasy'
>>> mpl.rcParams['font.fantasy'] = 'Comic Sans MS, Arial'
>>> plt.plot([1,2,3,4], [1,4,9,16], 'ro')
[<matplotlib.lines.Line2D object at 0x000001BB6E03B470>]
>>> plt.xlabel('Значення x')
Text(0.5,0,'Значення x')
>>> plt.ylabel('Значення x у квадраті')
Text(0,0.5,'Значення x у квадраті')
>>> plt.show()
```

В результаті має вийти ось що:



Функції pyplot.plot() можна передати довільну кількість пар аргументів типів list або array (тип array визначений у модулі numpy).

У прикладі вище передано одну пару: [1,2,3,4], [1,4,9,16]. Перший елемент кожної пари функція розглядає як значення, що відкладаються по осі X, другий елемент як значення осі Y. Відповідно для кожної пари елементів будуватися своя крива на діаграмі.

Після кожної пари даних може бути переданий додатковий аргумент типу string – рядок форматування, який визначає зовнішній вигляд кривої.

Формат рядка запозичений із системи Matlab. Рядок включає три підрядки.

- перший підрядок задає колір графічного елемента,
- друга стиль маркера,
- третій стиль лінії.

У прикладі 'ro' = 'r' + 'o', де 'r' - червоний, 'o' - кружок. За замовчуванням 'b-' – синя безперервна лінія.

Архітектура matplotlib

Одне з основних завдань, що виконує matplotlib - надання набору функцій та інструментів для представлення та управління Figure (так називається основний об'єкт) разом із усіма внутрішніми об'єктами, з якого він складається. Але в matplotlib є інструменти для обробки подій і, наприклад, анімації. Завдяки їм ця бібліотека здатна створювати інтерактивні графіки на основі подій натискання кнопки або руху миші.

Архітектура matplotlib логічно розділена на три шари, розташовані на трьох рівнях. Комунікація непряма - кожен шар може взаємодіяти тільки з тим, що розташований під ним, але не над.

Ось ці шари:

- Шар сценарію
- Художній шар
- Шар бекенда

Шар бекенда

Шар Backend є нижнім на діаграмі з архітектурою усієї бібліотеки. Він містить усі API та набір класів, які відповідають за реалізацію графічних елементів на низькому рівні.

- `FigureCanvas`- Це об'єкт, що втілює область малювання.
- `Renderer`- Об'єкт, який малює по FigureCanvas.
- `Event`- об'єкт, що обробляє введення від користувача (події з клавіатури та миші)

Художній шар

Середнім шаром є художній (artist). Усі елементи, що становлять графік, такі як назва, мітки осей, маркери тощо, є екземплярами цього об'єкта. Кожен їх грає свою роль ієрархічної структурі.

Є два мистецькі класи: примітивний та складовий.

- Примітивний - це об'єкти, які є базовими елементами для формування графічного представлення графіка, наприклад, Line2D, або геометричні фігури, такі як прямокутник коло або навіть текст.
- Складові - об'єкти, що складаються з кількох базових (примітивних). Це осі, шкали та діаграми.

На цьому рівні часто доводиться мати справу з об'єктами, що займають високе становище в ієрархії: графік, система координат, осі. Тому важливо повністю розуміти, яку роль вони відіграють. Нижче наведено три основні художні (складові об'єкти), які часто використовуються на цьому рівні.

- `Figure`- Об'єкт, що займає верхню позицію в ієрархії. Він відповідає всьому графічному уявленню і може містити багато систем координат.
- `Axes`— це цей графік. Кожна система координат належить лише одному об'єкту Figure і має два об'єкти Axis (або три, якщо йдеться про тривимірний графік). Інші об'єкти, такі як назва, мітки x та y, належать окремо осям.
- `Axis`враховує числові значення в системі координат, визначає межі та керує позначеннями на осях, а також відповідним кожному з них текстом. Положення шкал визначається об'єктом Locator, а зовнішній вигляд – Formatter.

Шар сценарію (pyplot)

Художні класи та пов'язані з ними функції (API matplotlib) підходять усім розробникам, особливо тим, хто працює із серверами веб-додатків або розробляє графічні інтерфейси. Але для обчислень, зокрема дляаналізу та

візуалізації данихнайкраще підходить шар сценарію. Він включає інтерфейс pyplot.

pylab і pyplot

З погляду користувача існують дві бібліотеки: pylab та pyplot. Pylab – це модуль, що встановлюється разом з matplotlib, а pyplot – внутрішній модуль matplotlib. Обидва часто посилаються в скриптах:

```
від pylab import *

# і

import matplotlib.pyplot as plt
import numpy as np
```

Pylab поєднує функціональність pyplot із можливостями NumPy в одному просторі імен, тому окремо імпортувати NumPy не потрібно. Більше того, при імпорті pylab функції з pyplot та NumPy можна викликати без посилання на модуль (простір імен).

```
plot(x,y)
array([1,2,3,4])
# замість
plt.plot()
np.array([1,2,3,4]
```

Пакет pyplot пропонує класичнийінтерфейс Pythonдля програмування, має власний простір має та потребує окремого імпорту NumPy. У наступних матеріалах використається цей підхід. Його ж застосовує більшість програмістів на Python.

Призначення клавіш інтерактивного вікна діаграми.

Кнопка

(pan/zoom) призначена для прокручування/масштабування діаграми.

Натискаємо на кнопку, покажчик миші набуває вигляду перехрещених стрілок.

Якщо рухати мишу і утримувати ліву кнопку - діаграма прокручуватиметься у напрямку руху покажчика.

Якщо рухати мишу і утримувати праву кнопку – масштаб діаграми буде змінюватися, при русі вгору/вправо – зменшуватись, вниз/вліво – збільшуватись.

При натиснутих клавішах "x" та "y" переміщення/масштабування діаграми відбуватиметься по осях X та Y відповідно.

При натиснутій кнопці Ctrl діаграма буде змінюватися так, щоб збереглися її пропорції (aspect ratio).

Кнопка

при натисканні включає режим прямокутного масштабування.

Якщо натиснути кнопку вказівник миші набуде вигляду хреста.

При натиснутій лівій кнопці миші на діаграмі можна виділити прямокутну область. Після звільнення кнопки масштаб діаграми буде змінено так, щоб виділена область зайняла якомога більшу площу діаграми.

Кнопки

дозволяють переміщатися з історії зміни діаграми. Всі зміни, що вносяться в діаграму користувачем (масштабування, прокручування), запам'ятовуються.

Кнопки зі стрілками дозволяють переміщатися вперед/назад за варіантами діаграми.

Кнопка з будиночком повертає діаграму до початкового стану.

Кнопка

викликає стандартний системи діалог збереження файла.

Дозволяє зберегти діаграму у файлі. Можна вибрати такі формати діаграми: png, pdf, svg, eps, ps.

У всіх прикладах імпортуємо модулі:

```
import matplotlib as mpl
# де-факто стандарт виклику pyplot у python
import matplotlib.pyplot as plt
```