

Python和科学计算基础

黄书剑



SciPy生态系统



- **SciPy (pronounced “Sigh Pie”) is a Python-based ecosystem of open-source software for mathematics, science, and engineering.**



NumPy
Base N-dimensional
array package



SciPy library
Fundamental library for
scientific computing



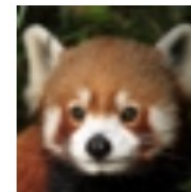
Matplotlib
Comprehensive 2-D
plotting

IP[y]:
IPython

IPython
Enhanced interactive
console

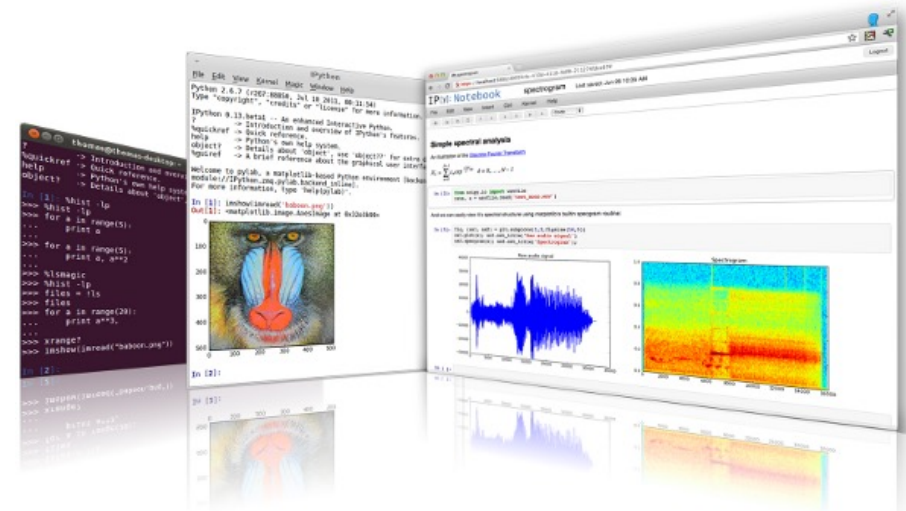


SymPy
Symbolic mathematics



pandas
Data structures &
analysis

- **IPython provides a rich architecture for interactive computing with:**
 - A powerful interactive shell.
 - A kernel for Jupyter.
 - Support for interactive data visualization and use of GUI toolkits.
 - Flexible, embeddable interpreters to load into your own projects.
 - Easy to use, high performance tools for parallel computing.



IP[y]:
IPython

- **NumPy is the fundamental package for scientific computing in Python.**
 - **a multidimensional array object**
 - various derived objects (such as masked arrays and matrices)
 - an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

- The SciPy library is one of the core packages that make up the SciPy stack. It provides many user-friendly and efficient numerical routines:

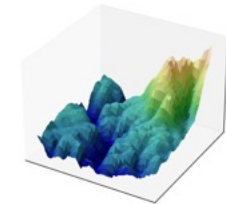
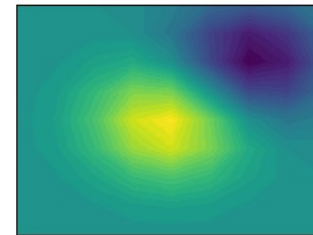
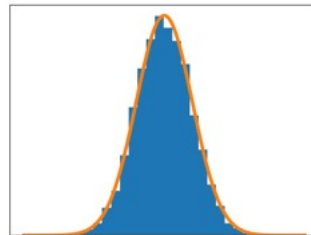
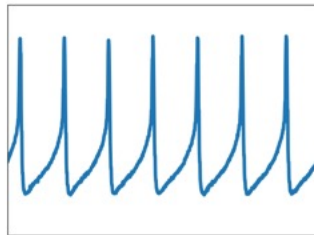
- numerical integration
- interpolation
- optimization
- linear algebra
- statistics.

Subpackage	Description
cluster	Clustering algorithms
constants	Physical and mathematical constants
fftpack	Fast Fourier Transform routines
integrate	Integration and ordinary differential equation solvers
interpolate	Interpolation and smoothing splines
io	Input and Output
linalg	Linear algebra
ndimage	N-dimensional image processing
odr	Orthogonal distance regression
optimize	Optimization and root-finding routines
signal	Signal processing
sparse	Sparse matrices and associated routines
spatial	Spatial data structures and algorithms
special	Special functions
stats	Statistical distributions and functions



Matplotlib: Visualization with Python

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.



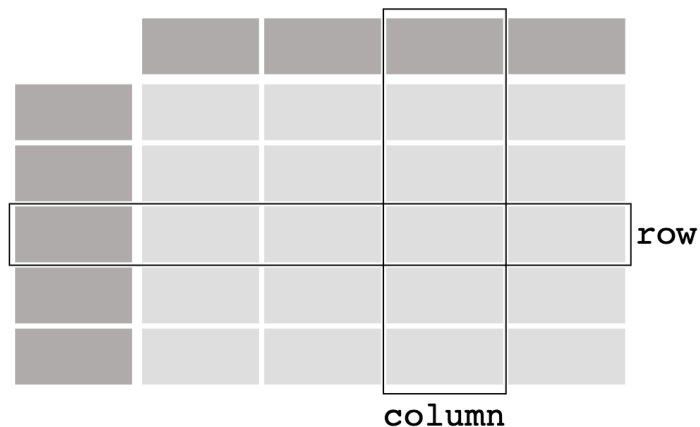
Matplotlib makes easy things easy and hard things possible.

Pandas

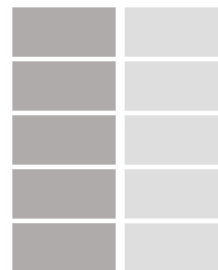


- pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive.

DataFrame



Series



A screenshot of a LibreOffice Calc spreadsheet titled "Untitled 1 - LibreOffice Calc". The spreadsheet displays a table with the following data:

	A	B	C	D	E	F
1		Name	Age	Sex		
2	0	Braund, Mr. Owen Harris	22	male		
3	1	Allen, Mr. William Henry	35	male		
4	2	Bonnell, Miss. Elizabeth	58	female		
5						
6						
7						
8						

The spreadsheet interface includes a menu bar (File, Edit, View, Insert, Format, Styles, Sheet, Data, Tools, Window, Help), a toolbar, and a status bar at the bottom showing "Sheet 1 of 1", "Default", "English (USA)", and "Average: ; Sum: 0".

SymPy



- SymPy is a Python library for symbolic mathematics.
- It aims to become a full-featured computer algebra system (CAS) while keeping the code as simple as possible in order to be comprehensible and easily extensible.

SymPy:

```
exp (x)/(1+exp (2*x))
```

$$\frac{e^x}{e^{2x} + 1}$$

SymPy:

```
23*pi +factorial (20)/(3**5)-2
```

$$23\pi + 10011942420479998 \approx 1.00119424204801 \cdot 10^{16}$$



IPYTHON环境

- 启动IPython控制台

```
(base) iMacS2:~ huangshujian$ ipython
Python 3.7.4 (default, Aug 13 2019, 15:17:50)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.8.0 -- An enhanced Interactive Python. Type '?' for
help.
```

```
In [1]:
```



主要特点

- 输入输出缓存 In Out
- 增强交互辅助功能
 - 语法高亮、自动补全
- 系统命令交互！
- Magic functions %
- 调试器、并行计算扩展等



带编号的输入输出

- 用于记录和输出

```
In [1]: 3 * 3
```

```
Out[1]: 9
```

```
In [2]: In[1]
```

```
Out[2]: '3 * 3'
```

```
In [3]: print('Hello World')
```

```
Hello World
```

```
In [4]: x = 5
```



带编号的输入输出

- 所有历史输入被组织为一个列表 In
- 所有历史输出被组织为一个字典 Out

```
In [5]: In
```

```
Out[5]: ['', '3 * 3', 'In[1]', "print('Hello World')", 'x = 5', 'In']
```

```
In [6]: Out
```

```
Out[6]:
```

```
{1: 9,  
 2: '3 * 3',  
 5: ['', '3 * 3', 'In[1]', "print('Hello World')", 'x = 5', 'In',  
    'Out']}
```



增强交互辅助功能

- 定义复合语句时，自动提示输入后续内容
- 语法高亮，自动补全等

```
In [8]: def print_func(x):  
...:     x = x + 5  
...:     print(x)  
...:
```

```
In [9]: print_func(100)  
105
```

获取帮助



- 快速了解主要功能和查阅文档

command	description
?	Introduction and overview of IPython's features.
%quickref	Quick reference.
help	Python's own help system.
object?	Details about 'object', use 'object??' for extra details.



- 查看对象的文档注释

```
In [13]: list?
```

```
Init signature: list(iterable=(), /)
```

```
Docstring:
```

```
Built-in mutable sequence.
```

If no argument is given, the constructor creates a new empty list.
The argument must be an iterable if specified.

```
Type:          type
```

```
Subclasses:    _HashedSeq, StackSummary, SList, _ImmutableLineList,  
FormattedText, NodeList, _ExplodedList, Stack, _Accumulator
```




与系统shell交互

- **！开头的命令将被解析为调用系统shell**

```
In [24]: !pwd  
/Users/huangshujian
```

```
In [26]: !ls  
Applications      My Cloud          gensim-data  
Desktop           Overall           nltk_data  
Documents         PaperWithCitations  opt
```



与系统shell交互

- 将系统命令结果赋值给python变量

```
In [28]: file = !ls
```

```
In [29]: file
```

```
Out[29]:
```

```
['Applications',  
 'Desktop',  
 'Documents',  
 'Downloads',  
 'Library',  
 .....
```

- 使用python变量执行系统命令

```
In [30]: filename = "Working"
```

```
In [36]: !ls
```

```
$filename
```

```
Icon? mactex-20200407.pkg screen  
erhan10a.pdf python 名单.txt
```



Magic functions

- 一系列%开头的辅助命令，用于控制IPython环境和系统行为等
 - %开头为单行命令 (line magic)
 - %%开头为多行命令 (cell magic)
- The magic function system provides a series of functions which allow you to control the behavior of IPython itself, plus a lot of system-type features.

使用%magic 查看magic function的相关说明



Magic functions

- 一系列%开头的辅助命令，用于控制IPython环境和系统行为等
 - %开头为单行命令 (line magic)
 - %%开头为多行命令 (cell magic)
- 关于代码控制:
 - %run, %edit, %save, %macro, %recall, etc.
- 关于文件系统:
 - %ls, %pwd, %cd, %cp, %less, %writefile
- 关于缓存:
 - %load, %paste
-



- 通过? 和? ? 查看不同详细程度的文档资料

```
In [17]: %lsmagic?
```

```
Docstring: List currently available magic functions.
```

```
File:      ~/opt/anaconda3/lib/python3.7/site-  
packages/IPython/core/magics/basic.py
```

```
In [18]: %lsmagic??
```

```
Source:
```

```
    @line_magic
```

```
    def lsmagic(self, parameter_s=''):
```

```
        """List currently available magic functions."""
```

```
        return MagicsDisplay(self.shell.magics_manager, ignore=[])
```

```
File:      ~/opt/anaconda3/lib/python3.7/site-  
packages/IPython/core/magics/basic.py
```



In [16]: %lsmagic

Out[16]:

Available line magics:

%alias %alias_magic %autoawait %autocall %autoindent %automagic %bookmark %cat %cd %clear %colors %conda %config %cp %cpaste %debug %dhist %dirs %doctest_mode %ed %edit %env %gui %hist %history %killbgscripts %ldir %less %lf %lk %ll %load %load_ext %loadpy %logo %ff %logon %logstart %logstate %logstop %ls %lsmagic %lx %macro %magic %man %matplotlib %mkdir %more %mv %notebook %page %paste %pastebin %pdb %pdef %pdoc %pfile %pinfo %pinfo2 %pip %popd %pprint %precision %prun %psearch %psource %pushd %pwd %pycat %pylab %quickref %recall %rehashx %reload_ext %rep %rerun %reset %reset_selective %rm %rmdir %run %save %sc %set_env %store %sx %system %tb %time %timeit %unalias %unload_ext %who %who_ls %whos %xdel %xmode

Available cell magics:

%%! %%HTML %%SVG %%bash %%capture %%debug %%file %%html %%javascript %%js %%latex %%markdown %%perl %%prun %%pypy %%python %%python2 %%python3 %%ruby %%script %%sh %%svg %%sx %%system %%time %%timeit %%writefile



```
In [39]: %automagic  
Automagic is ON, % prefix IS NOT needed for line magics.
```

```
In [40]: pwd  
Out[40]: '/Users/huangshujian'
```

```
In [41]: cd  
/Users/huangshujian
```

```
In [42]: cd Working/python  
/Users/huangshujian/Working/python
```

使用%automagic可以省略开头的% 23



```
In [43]: %%writefile fib.py
...: def fib(N):
...:     """
...:     Return a list of the first N Fibonacci numbers.
...:     """
...:     f0, f1 = 0, 1
...:     f = [1] * N
...:     for n in range(1, N):
...:         f[n] = f0 + f1
...:         f0, f1 = f1, f[n]
...:
...:     return f
...:
...: print(fib(10))
...:
...:
```

Writing fib.py



```
In [44]: !python fib.py  
[1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
```

可以通过系统命令执行

```
In [45]: run fib.py  
[1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
```

可以通过magic执行

```
In [46]: fib(6)  
Out[46]: [1, 1, 2, 3, 5, 8]
```

magic执行过后，相当于导入了该脚本，脚本中的变量名可以在后续代码中使用



代码性能分析

- **%timeit**
 - 多次运行给定语句，并给出平均运行时间
- **%time和%%time**
 - 对给定的line或cell进行运行计时
- **%prun**
 - 分析运行过程中不同部分的执行频率和时间
 - 利用profiler进行性能分析

更多性能分析内容参见：<https://docs.python.org/3/library/profile.html>



```
In [47]: %timeit fib(100)
```

```
9.71 µs ± 15.4 ns per loop (mean ± std. dev. of 7 runs, 100000  
loops each)
```

```
In [48]: %time fib(100)
```

```
CPU times: user 14 µs, sys: 0 ns, total: 14 µs
```

```
Wall time: 14.8 µs
```

```
Out[48]:
```

```
[1,  
 1,  
 2,  
 3,  
 5,  
 8,
```

```
def improve(update, close, guess = 1):  
    while not close(guess):  
        guess = update(guess)  
        print(guess)  
    return guess  
def appr_equal(x, y, epsilon = 1e-10):  
    return abs(x - y) < epsilon
```





```
def improve(update, close, guess = 1):  
    while not close(guess):  
        guess = update(guess)  
        print(guess)  
    return guess
```

```
def my_sqrt(x):  
    def sqrt_update(guess):  
        mid = average(guess)  
        if mid * mid > x :  
            return guess[0], mid  
        else :  
            return mid, guess[1]  
    def average(tup):  
        return (tup[0] + tup[1]) / 2  
    def sqrt_accurate(guess):  
        mid = average(guess)  
        return appr_equal(mid * mid, x)  
    return average(improve(sqrt_update, sqrt_accurate, (0, x)))
```



```
%prun print(my_sqrt(2))
```

```
214 function calls in 0.000 seconds
```

```
Ordered by: internal time
```

ncalls	tottime	percall	cumtime	percall	filename:lineno(function)
30	0.000	0.000	0.000	0.000	{built-in method builtins.print}
30	0.000	0.000	0.000	0.000	<ipython-input-54-9121da0881e4>:10(sqrt_accurate)
1	0.000	0.000	0.000	0.000	{built-in method builtins.exec}
1	0.000	0.000	0.000	0.000	<ipython-input-53-56a84c2e6d1e>:1(improve)
29	0.000	0.000	0.000	0.000	<ipython-input-54-9121da0881e4>:2(sqrt_update)
30	0.000	0.000	0.000	0.000	<ipython-input-53-56a84c2e6d1e>:6(appr_equal)
60	0.000	0.000	0.000	0.000	<ipython-input-54-9121da0881e4>:8(average)
30	0.000	0.000	0.000	0.000	{built-in method builtins.abs}
1	0.000	0.000	0.000	0.000	<ipython-input-54-9121da0881e4>:1(my_sqrt)
1	0.000	0.000	0.000	0.000	{method 'disable' of '_lsprof.Profiler' objects}
1	0.000	0.000	0.000	0.000	<string>:1(<module>)

```
%debug my_sqrt(2)
```



Jupyter Notebook

- 基于Client-Server模式的交互式运行环境
 - 以cell的形式组织和运行代码
 - 将控制台运行结果通过网页进行更好的呈现
 - 方便编辑修改代码并查看运行结果
 - 支持markdown、latex等编辑功能
 - 可以显示图片等多媒体资源
 - 支持代码和运行结果保存和导出
 - html、pdf等



注销

文件 编辑 查看 插入 单元格 服务 Widgets 帮助

Trusted



Python 3



```
In [1]: print('Hello World')
```

```
Hello World
```

```
In [2]: 2 * 3.14
```

```
Out[2]: 6.28
```

```
In [3]: x = 5
```

```
In [4]: x
```

```
Out[4]: 5
```

```
In [5]: In
```

```
Out[5]: ['', "print('Hello World')", '2 * 3.14', 'x = 5', 'x', 'In']
```

```
In [6]: Out
```

```
Out[6]: {2: 6.28,
         4: 5,
         5: ['', "print('Hello World')", '2 * 3.14', 'x = 5', 'x', 'In', 'Out']}
```


基础运行环境配置



利用conda系统管理运行环境

- 本部分内容需要安装的包: ipython, notebook
- 后续部分根据进程依次需要安装: numpy, scipy, matplotlib等
- 如果完全安装anaconda, 上述基础包应该已经安装在环境中
- 如果安装的是miniconda等原因, 无上述包, 可以利用conda方便的进行环境配置
 - conda update conda
 - conda create --name scipy-basic
 - conda activate scipy-basic
 - conda install PACKAGENAME

<https://docs.conda.io/projects/conda/en/latest/user-guide/cheatsheet.html>

(base) MBP2SJ:~ huangshujian\$ conda update conda

更新conda环境

Package Plan

environment location: /Users/huangshujian/miniforge3

added / updated specs:

- conda

The following packages will be downloaded:

提示下载的package

package	build		
conda-package-handling-1.7.3	py39h5161555_0	1.5 MB	conda-forge
cryptography-3.4.7	py39h73257c9_0	792 KB	conda-forge

...

The following packages will be UPDATED:

提示更新的package

conda-package-han~	1.7.2-py39h51e6412_0 --> 1.7.3-py39h5161555_0
cryptography	3.4.4-py39h6e07874_0 --> 3.4.7-py39h73257c9_0

...

Proceed ([y]/n)?

提示进行



此处显示当前正在使用的环境，默认为base

(base) MBP2SJ:~ huangshujian\$ conda env list

查看系统中所有环境

conda environments:

#

base * /Users/huangshujian/miniforge3



```
(base) MBP2SJ:~ huangshujian$ conda create --name scipy-basic
```

```
Collecting package metadata (current_repodata.json): done
```

```
Solving environment: done
```

创建一个新环境 `scipy-basic`

```
## Package Plan ##
```

```
environment location: /Users/huangshujian/miniforge3/envs/scipy-basic
```

```
Proceed ([y]/n)?
```

```
Preparing transaction: done
```

```
Verifying transaction: done
```

```
Executing transaction: done
```

```
#
```

```
# To activate this environment, use
```

```
#
```

```
#     $ conda activate scipy-basic
```

```
#
```

```
# To deactivate an active environment, use
```

```
#
```

```
#     $ conda deactivate
```



```
(base) MBP2SJ:~ huangshujian$ conda activate scipy-basic    切换至环境scipy-basic
(scipy-basic) MBP2SJ:~ huangshujian$ conda list             查看当前环境中的包列表
# packages in environment at /Users/huangshujian/miniforge3/envs/scipy-basic:
#
# Name                                Version                                Build  Channel
#
```

此处显示当前正在使用的环境，已切换至scipy-basic



```
(scipy-basic) MBP2SJ:~ huangshujian$ conda install ipython
```

安装ipython包

```
...
```

```
(scipy-basic) MBP2SJ:~ huangshujian$ ipython
```

启动ipython交互环境

```
...
```

```
(scipy-basic) MBP2SJ:~ huangshujian$ conda install notebook
```

安装notebook包

```
...
```

```
(scipy-basic) MBP2SJ:~ huangshujian$ jupyter notebook
```

启动jupyter notebook

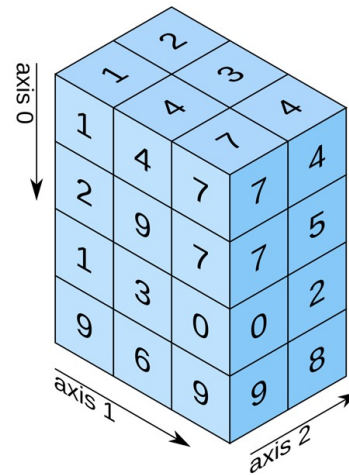
```
...
```

- **Python科学计算环境**
 - IPython环境、NumPy、SciPy、matplotlib、SymPy、Pandas
- **IPython环境使用**
 - magic functions
 - 代码性能分析
 - jupyter notebook
- **部分内容参考：《Python科学计算和数据科学应用》**

NUMPY

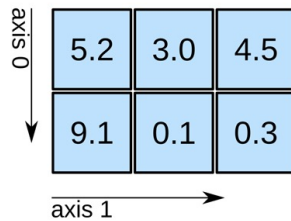
multi-dimensional array

3D array



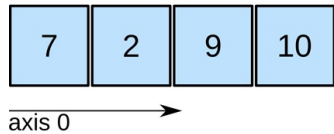
shape: (4, 3, 2)

2D array



shape: (2, 3)

1D array



shape: (4,)

$$\begin{array}{|c|} \hline \text{data} \\ \hline 1 \\ 2 \\ \hline \end{array} - \begin{array}{|c|} \hline \text{ones} \\ \hline 1 \\ 1 \\ \hline \end{array} = \begin{array}{|c|} \hline 0 \\ 1 \\ \hline \end{array}$$

$$\begin{array}{|c|} \hline \text{data} \\ \hline 1 \\ 2 \\ \hline \end{array} * \begin{array}{|c|} \hline \text{data} \\ \hline 1 \\ 2 \\ \hline \end{array} = \begin{array}{|c|} \hline 1 \\ 4 \\ \hline \end{array}$$

$$\begin{array}{|c|} \hline \text{data} \\ \hline 1 \\ 2 \\ \hline \end{array} / \begin{array}{|c|} \hline \text{data} \\ \hline 1 \\ 2 \\ \hline \end{array} = \begin{array}{|c|} \hline 1 \\ 1 \\ \hline \end{array}$$

$$\begin{array}{|c|c|} \hline \text{data} \\ \hline 1 & 2 \\ 3 & 4 \\ 5 & 6 \\ \hline \end{array} .\text{max}() = 6$$

$$\begin{array}{|c|c|} \hline \text{data} \\ \hline 1 & 2 \\ 3 & 4 \\ 5 & 6 \\ \hline \end{array} .\text{min}() = 1$$

$$\begin{array}{|c|c|} \hline \text{data} \\ \hline 1 & 2 \\ 3 & 4 \\ 5 & 6 \\ \hline \end{array} .\text{sum}() = 21$$

https://numpy.org/doc/stable/user/absolute_beginners.html