

Python 简易教程 (for Solar Group)

简介

优势: 开源, 用户广, 用途广, 社区完善, 新手友好

<https://www.jetbrains.com/research/python-developers-survey-2017/>

2017年的 python 使用调查报告 <https://www.jetbrains.com/research/python-developers-survey-2017/>

- 参考网站

- 官方网站: <https://www.python.org/>
- stackoverflow <https://stackoverflow.com/questions/tagged/python>
- 官方文档 <https://docs.python.org>
- 廖雪峰的网站 (包含 python2.7 和 python3 的教程) <https://www.liaoxuefeng.com>
- wikibooks https://en.wikibooks.org/wiki/Python_Programming
- 演示代码执行过程的工具 <http://pythontutor.com/>
- 天文相关教程(较新) <https://python4astronomers.github.io>

- 获取 python 包/模块/函数的帮助:

```
# python
>>> help(name)
```

```
# shell
$ pydoc <name>
$ pydoc -w <name> # 生成一个 html 帮助文件
```

```
# ipython/jupyter
name? # 弹出帮助文档, `Esc` 退出文档
```

基本语法

语法特点

- 对象编程的概念

类 -> 创建一个实例 -> 使用该实例所属的类下面的方法

```
l = [3, 2, 1] # l 是 list 的一个实例
l.reverse() # reverse 是一个方法(函数), reverse() 是一个实例
l.__dir__() # 查看 l 的所有方法(函数)
```

- 严格缩进(通常采用4个半角空格)

```
if a > 0:
    a += 1
print(a)
```

- 空格不敏感

```
a + b * c ** 2 / d
f (a, b)
```

- 大小写敏感, 使用大小写组合的变量名风格(下划线仍是合法变量名字符,但有特殊含义, 见 [python进阶#脚本相关](#))

```
from scipy.interpolate import BSpline
```

```
import pandas as pd
dataFrame = pd.DataFrame(array)
```

见 [IDL sav 文件读取](#)

- 括号内换行不用换行符

- 单行注释 # ...

- 多行注释

函数或模块的开头的多行注释(将存储在 __doc__ 变量), 可以用 help() 打印

```
"""
这是一个
多行注释
"""
```

- 同一行不同语句之间可用分号分隔

```
x = 1; print(x)
```

- 列表和数组的索引默认从0开始

- 列表的最后一个元素可以加逗号

```
[1, 2,]
{'a' : 1, 'b' : 2,}
```

- etc.

print函数

```
a = 1.; b = 2.
```

```
print('a=%.3f, b=%.1e' % (a, b))
```

a=1.000, b=2.0e+00

```
print(f'a={a}, b={b:.3f}')
```

a=1.0, b=2.000

```
%%python2
print('xxx'),
print('yyy')
```

xxx yyy

```
%%python3
print('xxx', end=' ')
print('yyy')
```

xxx yyy

更多格式化参考 <https://pyformat.info/>

基本运算

运算符参考:

https://en.wikibooks.org/wiki/Python_Programming/Basic_Math

https://www.tutorialspoint.com/python/python_basic_operators.htm

```
a = 1
a += 1  # 没有 a++ 这种语法
a
```

2

```
3. ** 2  # 自动转换数据类型
```

9.0

```
1 < 2 != 3  # 判断, 可以连写
```

True

```
a = None
a is None  # 注意这时不用 `==`
```

True

字符串

用双引号和单引号皆可(取决于所引内容).

字符串具有和列表(list)类似的运算规则(见之后的 list 说明):

```
'a' * 10
```

'aaaaaaaaaa'

```
'a' + 'b'
```

'ab'

```
'abc'[0]
```

'a'

```
print('\n')
```

```
print(r'\n')  # same as '\\n'
```

\n

list, tuple, dict, set

- 有序表 list, tuple

```
l = [0, 1, 2, 3, 4, 5] # list, 元素可变
t = (0, 1, 2, 3, 4, 5) # tuple, 元素不可变
```

```
l[0:3] # 一个切片(slice)
```

```
[0, 1, 2]
```

```
t[0:3]
```

```
(0, 1, 2)
```

```
l[1:5:2] # 索引: [start, stop, step] (结果不包含 stop)
```

```
[1, 3]
```

```
l[5:1:-1] # step < 0, 仍然不包含 stop (此例中 stop=1)
```

```
[5, 4, 3, 2]
```

list 常用运算

```
l = [0, 1, 2, 3, 4, 5]
l.append('x') # 添加在最后
print(l)
l.insert(3, 'y') # 添加在 l[i] 之前
print(l)
print(l.pop(1)) # 删除某元素并返回该值
print(l)
```

```
[0, 1, 2, 3, 4, 5, 'x']
[0, 1, 2, 'y', 3, 4, 5, 'x']
1
[0, 2, 'y', 3, 4, 5, 'x']
```

```
print(l * 2) # 重复N遍
print(l + [6, 7]) # 合并两个list
```

```
[1, 'y', 2, 3, 4, 5, 'x', 1, 'y', 2, 3, 4, 5, 'x']
[1, 'y', 2, 3, 4, 5, 'x', 6, 7]
```

list 可以排序(当元素为同一类型且可比较大小时), 返回到原列表

```
l = ['b', 'a']
l.sort()
l
```

```
['a', 'b']
```

```
l = [2, 1]
l.sort()
l
```

```
[1, 2]
```

• 字典 dict

```
d = {'a': 1, 'b': 2}
```

```
d = dict(a=1, b=2) # 注意这时没有引号
```

```
d['a']
```

```
1
```

• 集合 set

```
s = {3, 2, 1, 3} # set, 将自动排序并去除重复, 常用于 for 循环
s # 不能用 s[i] 来索引
```

```
{1, 2, 3}
```

set 不可索引, 但可用作 for 循环. 见 [python进阶#迭代器](#)

```
for i in s:  
    print(i)
```

```
1  
2  
3
```

```
l2 = [i for i in s]  
print(l2)
```

```
[1, 2, 3]
```

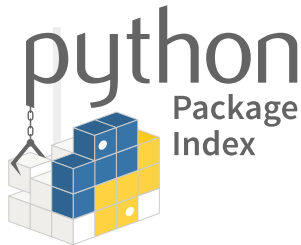
if 语句

python 里没有 case, 只有 if (非零数值、非空字符串、非空 list 等皆为 True)

```
if <条件判断1>:  
    <执行1>  
elif <条件判断2>:  
    <执行2>  
elif <条件判断3>:  
    <执行3>  
else:  
    <执行4>
```

相关工具

pip



The **Python Package Index (PyPI)** is a repository of software for the Python programming language.

PyPI helps you find and install software developed and shared by the Python community.

<https://pypi.org/>

```
$ pip install <pkg_name>
```

(其他命令见 [安装和配置#用 pip 安装 python 包](#))

Anaconda



Package, dependency and environment management for any language—Python, R, Ruby, Lua, Scala, Java, JavaScript, C/ C++, FORTRAN

Anaconda(<https://www.anaconda.com/>)

安装好的 conda 包含独立于系统的 python 版本, 一部分已编译好的包, pip 工具, 以及不同环境的子目录. 可根据需要切换不同的 conda 环境以使用不同的 python 版本或不同的包的组合.

```
$ conda install <pkg_name>
```

其他命令见 [安装和配置#使用 conda 命令](#)

详细说明见 [python进阶#Anaconda](#)

IPython & Jupyter

<http://jupyter.org/>



Project Jupyter exists to develop open-source software, open-standards, and services for interactive computing across dozens of programming languages.

<https://nbviewer.jupyter.org/>



参考文档 <http://jupyter-notebook.readthedocs.io/en/stable/>

- 打开 jupyter notebook:

```
$ jupyter notebook [filename]
```

常用包介绍

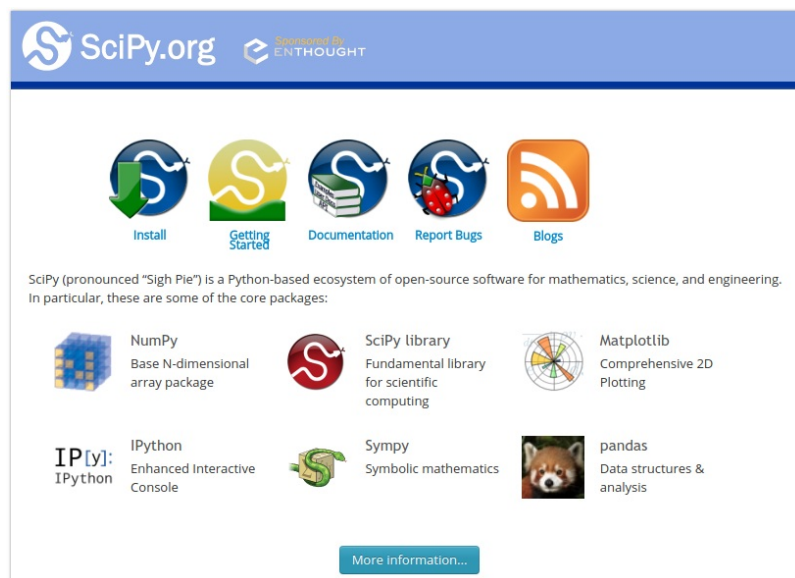
Python 自带包和模块

```
from math import * # 包含 pi 等, 但更推荐 numpy
from copy import deepcopy
from pprint import pprint # 打印 list, dict 等更好看
import requests
import os
import sys
```

NumPy, SciPy

概览







<https://www.scipy.org>



SciPy.org Sponsored by ENTHOUGHT

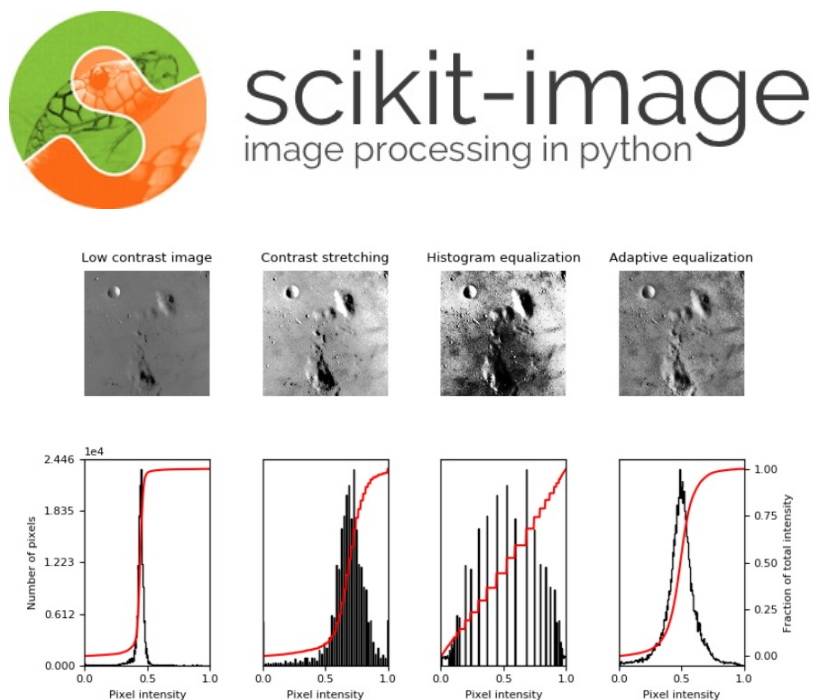
Install Getting Started Documentation Report Bugs Blogs

SciPy (pronounced "Sigh Pie") is a Python-based ecosystem of open-source software for mathematics, science, and engineering. In particular, these are some of the core packages:

 NumPy Base N-dimensional array package	 SciPy library Fundamental library for scientific computing	 Matplotlib Comprehensive 2D Plotting
 IPython Enhanced Interactive Console	 SymPy Symbolic mathematics	 pandas Data structures & analysis

[More information...](#)

<http://scikit-image.org>



```
import numpy as np
```

导入包的不同方式将在各例子中看到.

参考:

- NumPy User Guide <https://docs.scipy.org/doc/numpy/user/>
- NumPy Reference <https://docs.scipy.org/doc/numpy/reference/>
- ndarray <https://docs.scipy.org/doc/numpy/reference/arrays.ndarray.html>
- Array manipulation routines
<https://docs.scipy.org/doc/numpy/reference/routines.array-manipulation.html>
- SciPy Cookbook <http://scipy-cookbook.readthedocs.io/>
- 与 IDL 的语法对照
 - <http://mathesaurus.sourceforge.net/idl-numpy.html>
 - <http://blog.rtwilson.com/ten-little-idl-programs-in-python/>
 - <http://www.astrobetter.com/blog/2009/05/04/idl-vs-python/>

生成 numpy.ndarray 数组

将 list 或 tuple 转换为 numpy.ndarray

```
import numpy as np
a = np.array([0, 1, 2, 3, 4])
# 切片(slice), 索引可以是 int 或 list (python list 只支持 list作为索引)
a[0]          # 0
a[-1]         # 4
a[0:3]        # array([0, 1, 2]) 注意!
a[:3]         # array([0, 1, 2])
a[2:4]        # array([2, 3])
a[2:-1]       # array([2, 3]) 注意!
a[2:]         # array([2, 3, 4])
a[0:4:2]       # array([0, 2])
a[4:2:-1]     # array([4, 3])
a[[1, 4, 3]]  # array([1, 4, 3]) 花式切片, 索引是一个 list (不能是tuple)
```

多维:

```
a = np.array([[0, 1, 2], [3, 4, 5]]); a
array([[0, 1, 2],
       [3, 4, 5]])
```

```
a = np.array((0, 1, 2), (3, 4, 5)); a
# 这里的tuple已转成ndarray, 于是结果是可变的
array([[0, 1, 2],
       [3, 4, 5]])
```

```
a[0, 1]
1
```

```
a[0][1]
1
```

从一维数组转变为多维:

```
a = np.arange(12).reshape((3, -1)); a
# reshape 的参数为 < 0 表示该维度自动判断元素个数
# reshape 的参数是一个 list 或 tuple, 括号可以展开(详见帮助文档)
# 此例中 reshape((3, 2)), reshape(3, 2), reshape((3, -1)), reshape(3, -1) 等价
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

```
b = a.reshape(-1); b # reshape 的参数只有一个维度且 < 0(自动) => 展开成一维
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11])
```

花式切片的多维例子:

```
a = np.arange(12).reshape(3, -1)
a[[0, 1], [0, 2]]
# a[col_inds, row_inds], 此例等价于 a[[[0, 1], [0, 2]], a[[[0, 1], [0, 2]]]
array([0, 6])
```

- `np.zeros(shape, dtype=float)`, `np.ones(shape, dtype=float)`

```
np.zeros((2, 3)) # 默认浮点型
```

```
array([[0., 0., 0.],  
       [0., 0., 0.]])
```

```
np.ones((2, 3)) # 默认浮点型
```

```
array([[1., 1., 1.],  
       [1., 1., 1.]])
```

- **np.arange**([start,] stop[, step,], dtype=None)

生成的数组元素数据类型和参数一致.

方便记忆: 最简形式 `np.arange(N)` 中的整数 N 即为 元素个数.

```
np.arange(6).reshape((3, 2))
```

```
array([[0, 1],  
       [2, 3],  
       [4, 5]])
```

```
np.arange(6).reshape((3, -1))
```

```
array([[0, 1],  
       [2, 3],  
       [4, 5]])
```

- **np.linspace**(start, stop, num=50, endpoint=True, retstep=False, dtype=None)

默认生成的元素是浮点型

注意 `np.linspace` 和 `np.arange` 的不同规则

```
np.linspace(0, 1, 2)
```

```
array([0., 1.])
```

```
np.linspace(0, 1, 2, endpoint=False)
```

```
array([0. , 0.5])
```

更多数组创建方法见 [numpy 官网](https://docs.scipy.org/doc/numpy/reference/routines.array-creation.html)

<https://docs.scipy.org/doc/numpy/reference/routines.array-creation.html>

<https://docs.scipy.org/doc/numpy/user/basics.creation.html#arrays-creation>

- 转置

```
a = np.arange(12).reshape(3, -1); a
```

```
array([[ 0,  1,  2,  3],  
       [ 4,  5,  6,  7],  
       [ 8,  9, 10, 11]])
```

```
a.T
```

```
array([[ 0,  4,  8],  
       [ 1,  5,  9],  
       [ 2,  6, 10],  
       [ 3,  7, 11]])
```

更多数组操作见 [numpy 官网](https://docs.scipy.org/doc/numpy/reference/routines.array-manipulation.html) <https://docs.scipy.org/doc/numpy/reference/routines.array-manipulation.html> 矩阵操作见 [python 进阶#矩阵](#)

Matplotlib

- 官方示例 <https://matplotlib.org/tutorials/index.html>



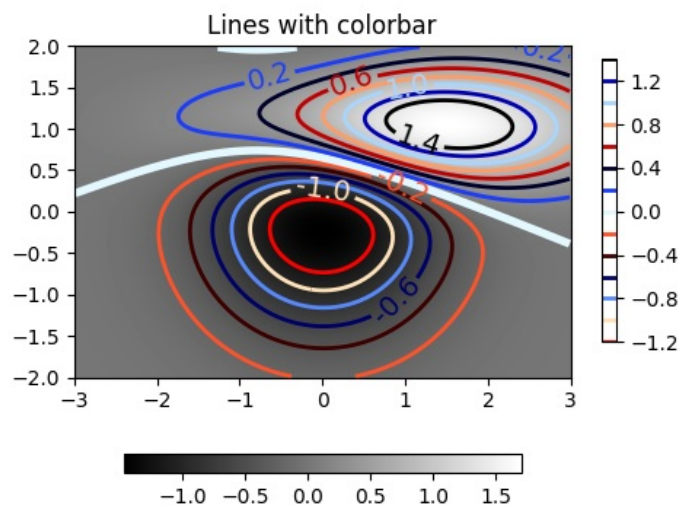
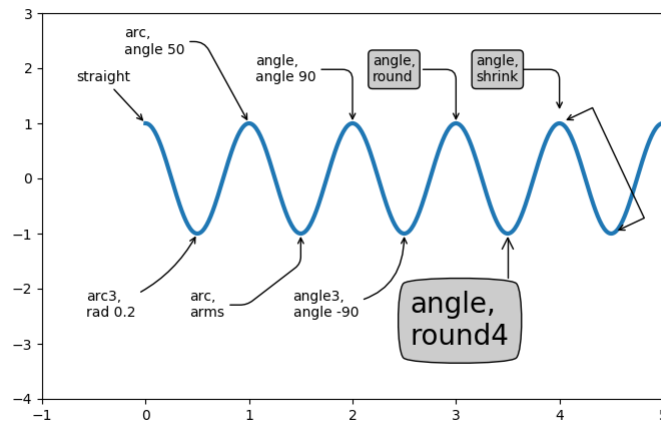
[home](#) | [examples](#) | [tutorials](#) | [pyplot](#) | [docs](#) »

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits.



Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc., with just a few lines of code. For examples, see the [sample plots](#) and [thumbnail gallery](#).

For simple plotting the `pyplot` module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.



- 官方教程 <https://matplotlib.org/tutorials/introductory/usage.html>

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

创建一个图像(三种写法皆可):

```
fig, ax = plt.subplots()
```

```
fig = plt.figure(1, (12, 6), dpi=100)
ax = fig.add_subplot(111) # 数字(ijk)表示: 这是一共i行j列中的组图中的第k个图
```

```
fig = plt.gcf()
ax = plt.gca()
```

获取帮助:

```
help(ax)
```

```
# 查看某对象包含的方法
ax.# <Tab>, <Shift-Tab>
plt.# <Tab>, <Shift-Tab>
```

例:

```
%matplotlib?
```

```
%matplotlib
%run -e 'examples/test_plot.py'
```

```
%matplotlib
%run -e 'examples/test_imshow.py'
```

- 颜色 https://matplotlib.org/examples/color/colormaps_reference.html
- legend https://matplotlib.org/users/legend_guide.html

black	k	dimgray	dimgray
gray	grey	darkgray	darkgrey
silver	lightgray	lightgrey	gainsboro
whitesmoke	w	white	snow
rosybrown	lightcoral	indianred	brown
firebrick	maroon	darkred	r
red	mistyrose	salmon	tomato
darksalmon	coral	orangered	lightsalmon
sienna	seashell	chocolate	saddlebrown
sandybrown	peachpuff	peru	linen
bisque	darkorange	burlywood	antiquewhite
tan	navajowhite	blanchedalmond	papayawhip
moccasin	orange	wheat	oldlace
floralwhite	darkgoldenrod	goldenrod	cornsilk
gold	lemonchiffon	khaki	palegoldenrod
darkkhaki	ivory	beige	lightyellow
lightgoldenrodyellow	olive	y	yellow
olivedrab	yellowgreen	darkolivegreen	greenyellow
chartreuse	lawngreen	honeydew	darkseagreen
palegreen	lightgreen	forestgreen	limegreen
darkgreen	g	green	lime
seagreen	mediumseagreen	springgreen	mintcream
mediumspringgreen	mediumaquamarine	aquamarine	turquoise
lightseagreen	mediumturquoise	azure	lightcyan
paleturquoise	darkslategray	darkslategrey	teal
darkcyan	c	aqua	cyan
darkturquoise	cadetblue	powderblue	lightblue
deepskyblue	skyblue	lightskyblue	steelblue
aliceblue	dodgerblue	lightslategray	lightslategrey
slateblue	slategrey	lightsteelblue	cornflowerblue
royalblue	ghostwhite	lavender	midnightblue
navy	darkblue	mediumblue	b
blue	slateblue	darkslateblue	mediumslateblue
mediumpurple	rebeccapurple	blueviolet	indigo
darkorchid	darkviolet	mediumorchid	thistle
plum	violet	purple	darkmagenta
m	fuchsia	magenta	orchid
mediumvioletred	deeppink	hotpink	lavenderblush
palevioletred	crimson	pink	lightpink

天文相关包

Astropy



官方文档 <http://docs.astropy.org/en/stable/index.html>

坐标系统 <http://docs.astropy.org/en/stable/coordinates/index.html>

```
from astropy import units as u
from astropy.coordinates import SkyCoord
```

参考 <https://python4astronomers.github.io/astropy/astropy.html>

SunPy

SunPy

About

Documentation

Blog

Support Us

Get Help

SunPy Project

SunPy

ndcube

drms

radiospectra

IRISPy

Search

SunPy Guide

Code Reference

Example Gallery

Developer's Guide

Release History

Code of Conduct

SunPy Documentation

Welcome to the SunPy documentation. SunPy is a community-dev analysis environment for Python.

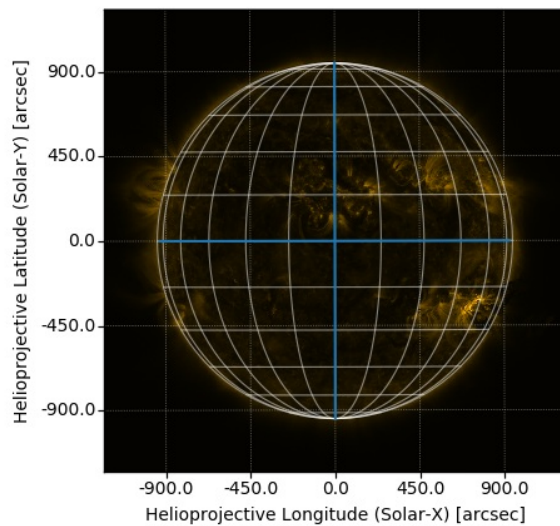
We have a documentation [index](#) and a [module](#) list.

- SunPy Guide**
 - Installation
 - A brief tour of SunPy
 - Acquiring Data with SunPy
 - Data Types in SunPy
 - Plotting in SunPy
 - Units and Coordinates in SunPy
 - Time in SunPy
 - Region of Interest
 - Customizing SunPy
 - SSWIDL/SunPy Cheat Sheet
 - Reporting Bugs
 - Testing New Features
 - Troubleshooting

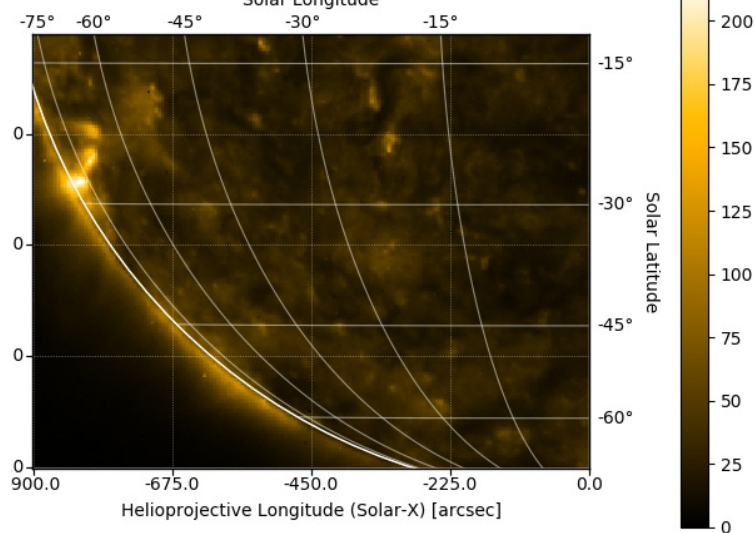
© 2019 The SunPy Community

[Github](#)
[Twitter](#)
[Mastodon](#)

AIA 171 Å 2011-06-07 06:33:02



SWAP 174 Å 2011-06-07 06:33:29
Solar Longitude



HMI 示例见 [python进阶#SunPy 示例](#)

<https://coding.net/u/lydiazly/p/scripts-sunpy/git?public=true>

- 参考

- Gallery <http://docs.sunpy.org/en/stable/generated/gallery/index.html>
- SunPy Guide <http://docs.sunpy.org/en/stable/guide/index.html>
- SunPy Map http://docs.sunpy.org/en/stable/code_ref/map.html#sunpy.map.mapbase.GenericMap
- Example Gallery <http://docs.sunpy.org/en/stable/generated/gallery/index.html>
- A brief tour of SunPy <http://docs.sunpy.org/en/stable/guide/tour.html>
- Finding and Downloading Data using Fido http://docs.sunpy.org/en/stable/guide/acquiring_data/fido.html

简单的例子(来自官网):

```
%matplotlib inline

import sunpy.map
import matplotlib.pyplot as plt
import sunpy.data.sample

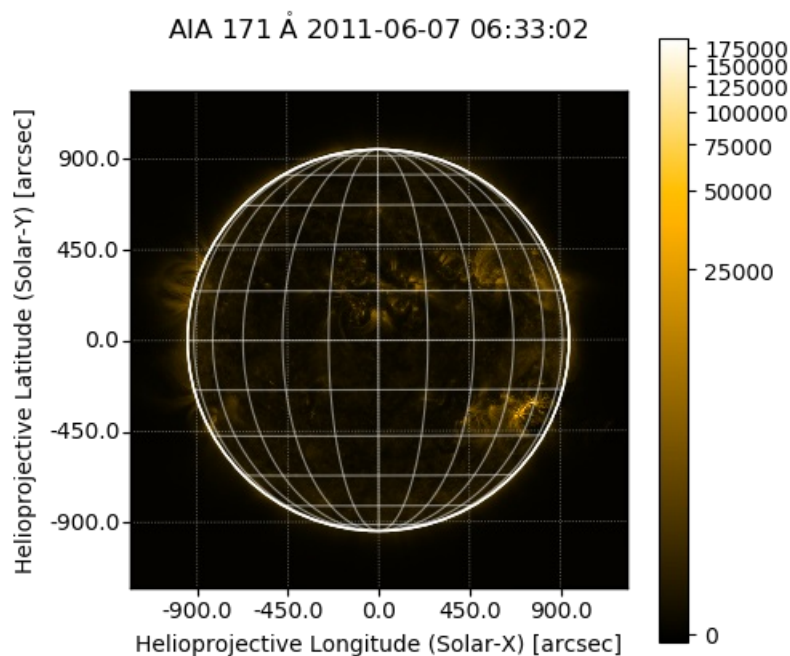
aia = sunpy.map.Map(sunpy.data.sample.AIA_171_IMAGE)

fig = plt.figure(1, (5, 5), dpi=100)
ax = plt.subplot(111, projection=aia)

aia.plot()
aia.draw_limb()
aia.draw_grid()
aia.draw_limb()
plt.colorbar();

# plt.show()

/home/lydia/miniconda3/lib/python3.6/site-packages/sunpy/map/sources/sdo.py:48: SunpyDeprecationWarning: Use Matplotlib to load the colormaps
  self.plot_settings['cmap'] = cm.get_cmap(self._get_cmap_name())
```



获取一些常数:

```
from sunpy.sun import constants as con
# one astronomical unit (the average distance between the Sun and Earth)
print(con.au)
```

```
Name      = Astronomical Unit
Value     = 149597870700.0
Uncertainty = 0.0
Unit      = m
Reference = IAU 2012 Resolution B2
```

```
# the solar radius
print(con.radius)
```

```
Name      = Solar radius
Value     = 695508000.0
Uncertainty = 26000.0
Unit      = m
Reference = Allen's Astrophysical Quantities 4th Ed.
```

常见数据文件读写

一般情况

```
f = open(filename, 'w') # 'w', 'r', ...
...
...
f.close()
```

更好的用法:

```
with open(filename, 'w') as f: # 'w', 'r', ...
    ...
    ...
```

txt, csv 文件读写

- 逐行读取整行的方式:
使用 `strip` 可以用来去掉读入的每行末尾的 `'\n'`

```
lines = []
with open('data/data.csv', 'r') as f:
    for line in f:
        lines.append(line.strip().split(','))
lines
```

```
[['col1', 'col2', 'col3'],
 ['row1', '1', '1e1'],
 ['row2', '2', '2e-1'],
 ['row3', '3', '3e+2']]
```

- 使用 `numpy`, 读取后直接转成数组的方式(推荐):
`np.loadtxt`, `np.genfromtxt` (写文件是 `np.savetxt`)
超大文件读取建议用 `pandas` 包.

e.g.

```
>>> import numpy as np
>>> arr = np.loadtxt(fname, skiprows=1, delimiter=',') # 默认分割符为空格
>>> col1, col2 = np.loadtxt(fname, skiprows=1, usecols=(1, 2), unpack=True)
```

`np.genfromtxt` 和 `np.loadtxt` 类似, 增加了处理 missing values 的功能.
(但参数 `skiprows=1` 需替换为 `skip_header=1`)

例 (这里使用功能更多的 `np.genfromtxt` 代替 `np.loadtxt`):

```
arr = np.genfromtxt('data/data.csv',
                    skip_header=1,
                    names=('col1', 'col2', 'col3'),
                    dtype=('U15', 'int', 'float'),
                    delimiter=',')
print(arr[['col1', 'col2']]) # 设定了 names 之后, 可以用名称索引

[('row1', 1) ('row2', 2) ('row3', 3)]
```

```
arr = np.genfromtxt('data/data2.csv',
                    skip_header=1,
                    names=('col1', 'col2', 'col3'),
                    dtype=('U15', 'int', 'float'),
                    missing_values=None,
                    filling_values=(' ', 0, np.nan), # 分开指定
                    delimiter=',')

print(arr)

[('row1', 0, 10. ) (' ', 2, 0.2) ('row3', 3, nan)]
```

- 一般而言, 对于一个由分隔符分隔的字符串, 转成一个数组的方式:
`np.fromstring(string, dtype=float, count=-1, sep='')`

```
np.fromstring('0 1 2', dtype=int, sep=' ')

array([0, 1, 2])
```

```
np.fromstring('0., 1., 2.', sep=',')

array([0., 1., 2.])
```

numpy save 文件读写

```
np.save('fname', arr) # 'fname.npy'
np.savez('fname', arr) # 'fname.npz', 多个变量
np.savez_compressed('fname', arr) # 'fname.npz', 多个变量 & 压缩
np.load('fname.npz') # Load the files created by savez_compressed.
```

```
# 存入
arr = np.ones(5) # 默认 `float64`
np.savez('data/np_array', a=arr, b=0)
```

```
ls data/*.npz

data/np_array.npz
```

```
# 载入
data = np.load('data/np_array.npz')
print((data['a'], data['b'], int(data['b']))) # 默认以`ndarray`读取

(array([1., 1., 1., 1., 1.]), array(0), 0)
```

IDL sav 文件读取

```
from scipy.io.idl import readsav
data = readsav('dname.sav', verbose=True, python_dict=False)
```

默认读出的是 `scipy.io.idl.AttrDict` 类型, 即 key 的大小写不敏感.

想要得到大小写敏感的 python dict, 设置 `python_dict=True` 即可.

例:

```

from scipy.io.idl import readsav
data = readsav('data/myidlfile.sav')
print(type(data))
print(data.keys()) # 注意返回的是一个`dict_keys`对象, 如需索引需要手动转为list
print(data['X'].shape) # 大小写不敏感
data = readsav('data/myidlfile.sav', python_dict=True)
print(type(data))
print(data['x'].shape) # 大小写敏感

<class 'scipy.io.idl.AttrDict'>
dict_keys(['x', 'y', 'str'])
(25,)
<class 'dict'>
(25,)

```

FITS 文件读取

- Astropy

参考:

<http://docs.astropy.org/en/stable/io/fits>

<https://python4astronomers.github.io/astropy/fits.html>

[Note]

If you are already familiar with PyFITS, astropy.io.fits is in fact the same code as the latest version of PyFITS, and you can adapt old scripts that use PyFITS to use Astropy by simply doing:

```
from astropy.io import fits as pyfits
```

However, for new scripts, we recommend the following import:

```
from astropy.io import fits
```

`fits.open`

```

>>> from astropy.io import fits
>>> hdulist = fits.open('<filename>.fit')
>>> hdulist.info() # hdulist 是一个由 HDU objects 组成的类似列表的对象
Filename: <filename>.fit
No.      Name          Type          Cards    Dimensions   Format
0       PRIMARY       PrimaryHDU    ...      ...         ...
1       ENERGIES      BinTableHDU  ...      ...         ...
>>> hdu = hdulist[0]
>>> hdu.data # ndarray
>>> hdu.header['<key>'] # 得到某个 header keyword
# 修改数据后
>>> hdu.writeto('<filename>.fits') # 保存 HDU object
>>> hdulist.writeto('<filename>.fits') # 或保存整个表

```

```

# 创建一个 fits Primary HDU object
>>> hdu = fits.PrimaryHDU()
>>> hdu.writeto('<filename>.fits') # 保存 HDU object
>>> hdu.writeto('<filename>.fits', clobber=True) # 保存到已存在文件

```

`fits.getdata, fits.getheader`

```

>>> data = fits.getdata('<filename>.fit')
>>> header = fits.getheader('<filename>.fit')

```

- SunPy

```
>>> import sunpy.io
>>> hdulist = sunpy.io.read_file('<filename>.fits') # 自动判断文件类型
# or use `hdulist = sunpy.io.fits.read('<filename>.fits')`
>>> hdu = hdulist[1]
>>> hdu.data # ndarray
```

```
>>> import sunpy.map
>>> smap = sunpy.map.Map('<filename>.fits')
# 得到一个 sunpy `GenericMap` object
```

例:

```
from astropy.io import fits
fname = 'scripts-sunpy/plot_hmi/data/hmi.B_720s.20150827_052400_TAI.field.fits'
hdulist = fits.open(fname)
```

```
hdulist.info()
```

```
Filename: scripts-sunpy/plot_hmi/data/hmi.B_720s.20150827_052400_TAI.field.fits
No.      Name      Ver      Type      Cards      Dimensions      Format
  0  PRIMARY          1 PrimaryHDU         6          ()
  1              1 CompImageHDU    155    (4096, 4096)    int32
```

```
hdulist.verify('silentfix+warn')
hdu = hdulist[1]
hdu.data.shape
```

```
(4096, 4096)
```

```
import sunpy.io
hdulist = sunpy.io.read_file(fname)
hdu = hdulist[1]
hdu.data.shape
```

```
(4096, 4096)
```

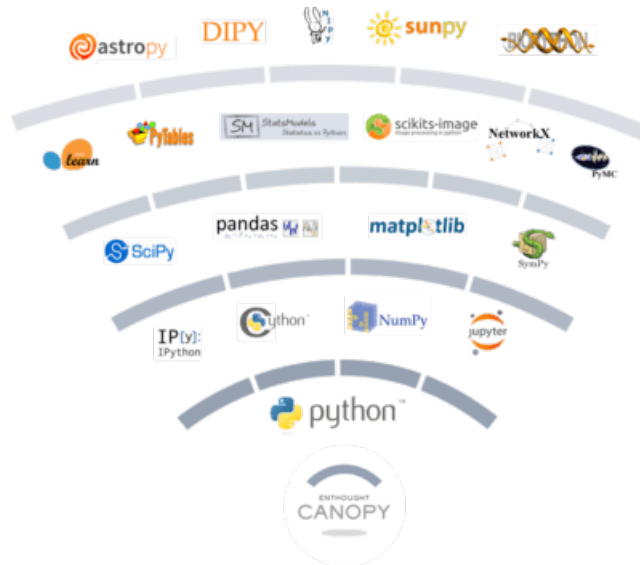
其他包和软件

- Astroconda <https://astroconda.readthedocs.io/en/latest/>

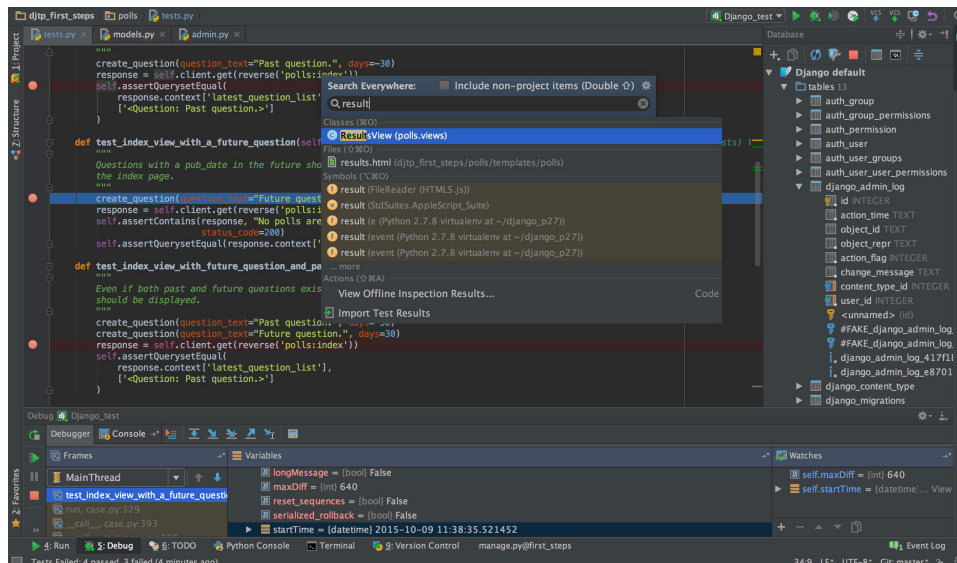
AstroConda is a free Conda channel maintained by the Space Telescope Science Institute (STScI) in Baltimore, Maryland. This channel provides tools and utilities required to process and analyze data from the Hubble Space Telescope (HST), James Webb Space Telescope (JWST), and others.

- Canopy <https://www.enthought.com/product/canopy/>

Enthought Canopy provides a proven scientific and analytic Python package distribution plus key integrated tools for iterative data analysis, data visualization, and application development. Users have the ability to extend and innovate with scripting and open platform APIs, driving the creation and sharing of innovative workflows, tools, and applications.



- PyCharm <https://www.jetbrains.com/pycharm/>



- Jupyterlab <http://jupyterlab.readthedocs.io/en/stable/>

File Edit View Run Kernel Tabs Settings Help

Python 3

In this Notebook we explore the Lorenz system of differential equations:

$$\begin{aligned}\dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xz \\ \dot{z} &= -\beta z + xy\end{aligned}$$

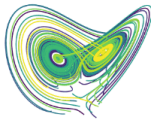
Let's call the function once to view the solutions. For this set of parameters, we see the trajectories swirling around two points, called attractors.

```
In [4]: from lorenz import solve_lorenz
t, x_t = solve_lorenz(N=10)
```

Output View

loreنز.py

sigma 10.00
beta 2.67
rho 28.00



```
def solve_lorenz(N=10, max_time=4.0, sigma=10.0, beta=8./3, rho=28.0):
    """Plot a solution to the Lorenz differential equations."""
    fig = plt.figure()
    ax = fig.add_axes([0, 0, 1, 1], projection='3d')
    ax.axis('off')

    # prepare the axes limits
    ax.set_xlim((-25, 25))
    ax.set_ylim((-35, 35))
    ax.set_zlim((5, 55))

    def lorenz_deriv(x,y,z, t0, sigma=sigma, beta=beta, rho=rho):
        """Compute the time-derivative of a Lorenz system."""
        x, y, z = x,y,z
        return (sigma * (y - x), x * (rho - z) - y, x * y - beta * z)

    # Choose random starting points, uniformly distributed from -15 to 15
    np.random.seed(1)
    x0 = -15 + 30 * np.random.random((N, 3))
```