

Scipy基础一

SV10



黄天羽
www.python123.org



Scipy的安装



Install



Getting
Started



Documentation



Report Bugs

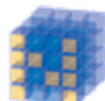


SciPy Central



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...](#)

[About SciPy](#)

[Install](#)

[Getting Started](#)

[Documentation](#)

[Bug Reports](#)

[Topical Software](#)

[Citing](#)

[SciPy Central](#) [↗](#)

[Cookbook](#) [↗](#)

[SciPy Conferences](#) [↗](#)

[Blogs](#) [↗](#)

[NumFOCUS](#) [↗](#)

CORE PACKAGES:

[Numpy](#) [↗](#)

[SciPy library](#) [↗](#)

[Matplotlib](#) [↗](#)

[IPython](#) [↗](#)

[SymPy](#) [↗](#)

SciPy的介绍

- 在Numpy库的基础上增加了众多的数学、科学以及工程计算中常用的库函数
- 例如：
 - 线性代数
 - 常微分方程数值求解
 - 信号处理
 - 图像处理
 - 稀疏矩阵
 -

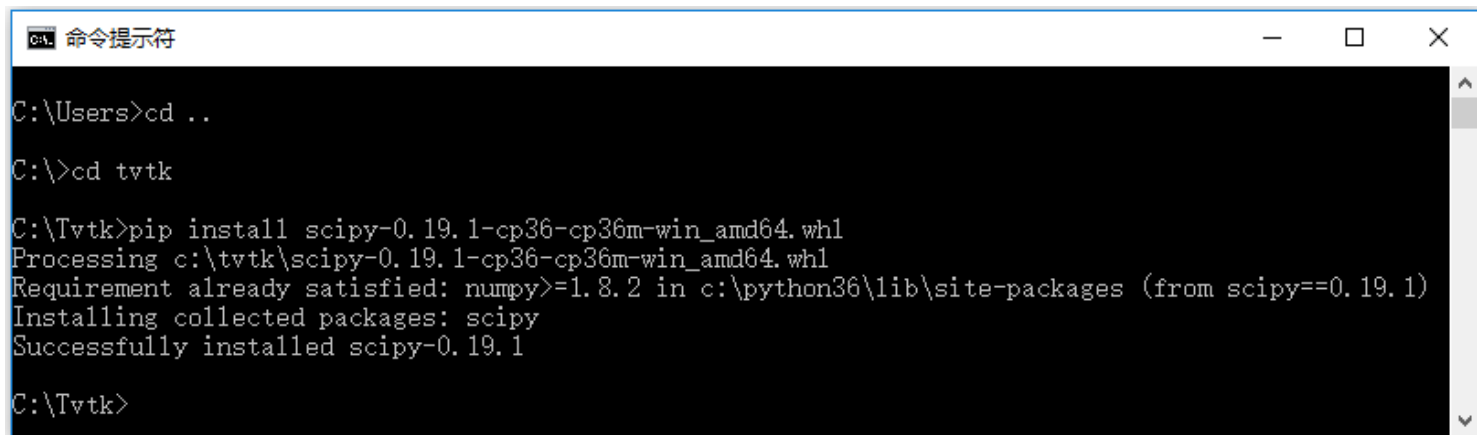
SciPy库的安装

Win平台：“以管理员身份运行” cmd

在下载目录执行

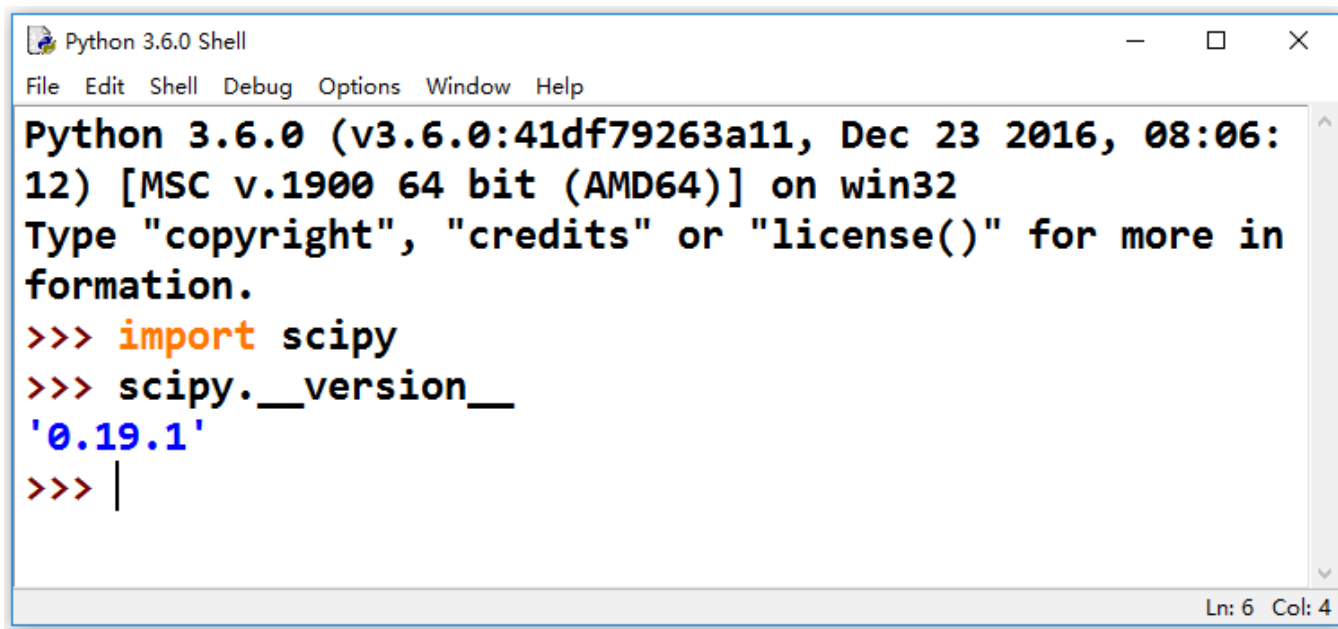
```
pip install numpy-1.13.0+mk1-cp36-cp36m-win_amd64.whl
```

```
pip install scipy-0.19.1-cp36-cp36m-win_amd64.whl
```

A screenshot of a Windows Command Prompt window titled "命令提示符". The window shows the following commands and output:

```
C:\Users>cd ..  
C:\>cd tvtk  
C:\Tvtk>pip install scipy-0.19.1-cp36-cp36m-win_amd64.whl  
Processing c:\tvtk\scipy-0.19.1-cp36-cp36m-win_amd64.whl  
Requirement already satisfied: numpy>=1.8.2 in c:\python36\lib\site-packages (from scipy==0.19.1)  
Installing collected packages: scipy  
Successfully installed scipy-0.19.1  
C:\Tvtk>
```

测试SciPy安装成功



A screenshot of a Python 3.6.0 Shell window. The window title is "Python 3.6.0 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area shows the following output and input:

```
Python 3.6.0 (v3.6.0:41df79263a11, Dec 23 2016, 08:06:12) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more in
formation.
>>> import scipy
>>> scipy.__version__
'0.19.1'
>>> |
```

The status bar at the bottom right indicates "Ln: 6 Col: 4".



SciPy的常数

SciPy的常数

SciPy的constants模块包含了众多的物理常数：

```
>>> from scipy import constants
>>> print(constants.c)
299792458.0
>>> print(constants.h)
6.62607004e-34
>>>
```

真空中的光速

普朗克常数

SciPy的常数

在字典`physical_constants`中，以物理常量名为键，对应的值是一个含有三个元素的元组，查看电子质量的例子：

```
>>> from scipy import constants
>>> constants.physical_constants["electron mass"]
(9.10938356e-31, 'kg', 1.1e-38)
>>> | 常数值      单位      误差
```

SciPy的常数

constants模块中还包含了许多单位信息，它们是1单元的量转换成标准单位时的数值：

```
>>> constants.mile      1英里等于多少米
1609.3439999999998
>>> constants.inch     1英寸等于多少米
0.0254
>>> constants.gram      1克等于多少千克
0.001
>>> constants.pound     1磅等于多少千克
0.45359236999999997
>>>
```



SciPy拟合与优化-optimize

SciPy的optimize模块

optimize模块提供了许多数值优化算法，可以实现

- 非线性方程组求解
- 数据拟合
- 函数最小值
-

最小二乘拟合

optimize库中的leastsq函数：对数据进行最小二乘拟合
调用形式为：

leastsq(func, x0)

func(x)是计算方程组误差的函数，它使得误差的平方和最小；x0为待确定参数的初始值。

最小二乘拟合-举例

假定有一组实验数据 (x_i, y_i) , 他们之间满足函数关系 $f(x) = kx + b$,
求解k和b是多少？

X	8.19, 2.72, 6.39, 8.71, 4.7, 2.66, 3.78
Y	7.01, 2.78, 6.47, 6.71, 4.1, 4.23, 4.05

最小二乘拟合-举例

则误差函数可以定义为：

```
import numpy as np
#计算以p为参数的直线和原始数据之间的误差
def f(p):
    k, b = p
    return (Y - (k*X+b))
```

最小二乘拟合-举例

求解和结果输出：

```
from scipy.optimize import leastsq
#leastsq使得f的输出数组的平方和最小，参数初始值为[1,0]
r = leastsq(f, [1,0])
k, b = r[0]
print("k=",k,"b=",b)
```


最小二乘拟合-举例

```
import numpy as np
from scipy.optimize import leastsq

X = np.array([8.19, 2.72, 6.39, 8.71, 4.7, 2.66, 3.78])
Y = np.array([7.01, 2.78, 6.47, 6.71, 4.1, 4.23, 4.05])

#计算以p为参数的直线和原始数据之间的误差
def f(p):
    k, b = p
    return(Y-(k*X+b))

#leastsq使得f的输出数组的平方和最小, 参数初始值为[1,0]
r = leastsq(f, [1,0])
k, b = r[0]
print("k=",k,"b=",b)
```

最小二乘拟合-举例

程序运行结果为：

```
k= 0.613495349193 b= 1.79409254326
```

```
>>>
```

非线性方程组求解

optimize库中的fsolve函数：对非线性方程组进行求解
调用形式为：

fsolve(func, x0)

func(x)是计算方程组误差的函数，它的参数x是一个矢量，表示方程组的各个未知数的一组可能解，func返回将x代入方程组之后得到的误差；x0为未知数矢量的初始值。

非线性方程组求解

对下面方程组进行求解：

$$f_1(u_1, u_2, u_3) = 0$$

$$f_2(u_1, u_2, u_3) = 0$$

$$f_3(u_1, u_2, u_3) = 0$$

误差函数func可以定义为：

```
def func(x):  
    u1,u2,u3 = x  
    return [f1(u1,u2,u3), f2(u1,u2,u3), f3(u1,u2,u3)]
```

非线性方程组求解-举例

使用fsolve求解非线性方程组：

$$\begin{cases} 5x_1 + 3 = 0 \\ 4x_0^2 - 2 \sin(x_1 x_2) = 0 \\ x_1 x_2 - 1.5 = 0 \end{cases}$$

非线性方程组求解-举例

则误差函数可以定义为：

```
from math import sin
def f(x):
    #转换为标准的浮点数列表
    x0, x1, x2 = x.tolist()
    return [5*x1+3,
            4*x0*x0 - 2*sin(x1*x2),
            x1*x2-1.5]
```

tolist()将x转换为Python的标准浮点数列表，在单个数值运算时，标准浮点数比NumPy的浮点类型更快，从而缩短计算时间

非线性方程组求解-举例

求解和结果输出：

```
from scipy.optimize import fsolve
#f是计算的方程组误差函数，[1,1,1]是未知数的初始值
result = fsolve(f, [1,1,1])
#输出方程组的解
print(result)
#输出误差
print(f(result))
```

非线性方程组求解-举例

```
from scipy.optimize import fsolve
from math import sin
#f计算方程组的误差,x是一组可能的解
def f(x):
    #转换为标准的浮点数列表
    x0, x1, x2 = x.tolist()
    return[5*x1+3,
           4*x0*x0 - 2*sin(x1*x2),
           x1*x2-1.5]
#[1,1,1]是未知数的初始值
result = fsolve(f, [1,1,1])
#输出方程组的解
print(result)
#输出误差
print(f(result))|
```

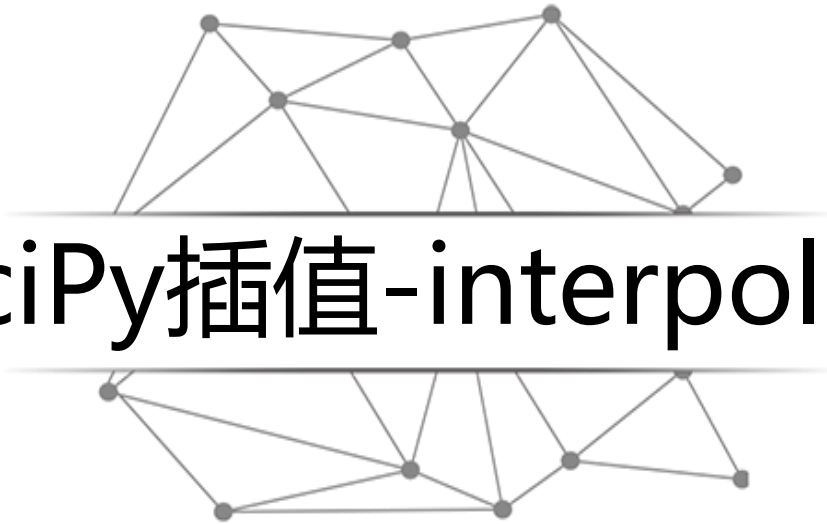

非线性方程组求解-举例

程序运行结果为：

非线性方程组求解结果

```
[-0.70622057 -0.6          -2.5          ]  
[0.0, -9.126033262418787e-14, 5.329070518200751e-15]  
>>>
```

方程组求解误差



SciPy插值-interpolate

插值 V.S. 拟合

插值：通过已知的离散数据来求解未知数据的方法，要求曲线通过所有的已知数据。

拟合：要求曲线函数与已知数据集的误差最小，不要求曲线通过所有的已知数据。

SciPy的interpolate模块

intepolate模块提供了许多对数据进行插值运算的函数：

- B样条曲线插值
- 外推
- Spline拟合（UnivariateSpline插值运算）
- 二维插值运算等
-

B样条曲线插值

一维数据的插值运算可以通过 `interp1d()` 实现。

其调用形式为：

`interp1d(x, y, kind='linear', ...)`

`Interp1d` 可以计算 `x` 的取值范围之内任意点的函数值，并返回新的数组。

参数 `x` 和 `y` 是一系列已知的数据点

参数 `kind` 是插值类型，可以是字符串或整数

B样条曲线插值

Kind给出了B样条曲线的阶数：

- 'zero' 'nearest' : 阶梯插值，相当于0阶B样条曲线
- 'slinear' 'linear' : 线性插值，相当于1阶B样条曲线
- 'quadratic' 'cubic' : 2阶和3阶B样条曲线，更高阶的曲线
可以直接使用整数值来指定
-

B样条曲线插值-举例

创建数据集：

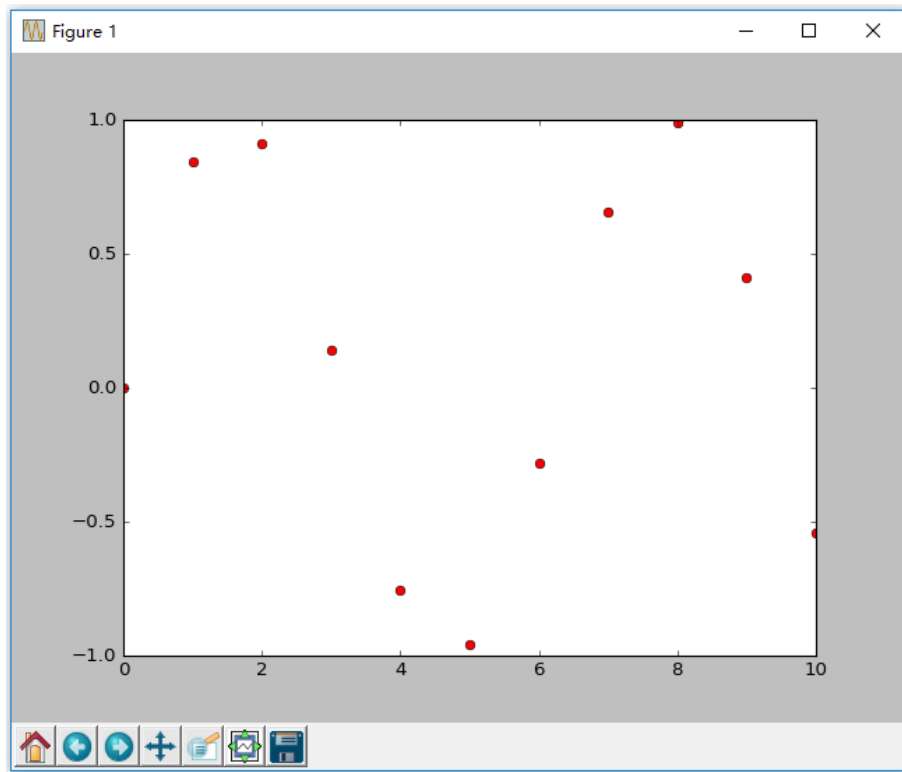
```
from numpy import np
x = np.linspace(0, 10, 11)
y = np.sin(x)
```

绘制数据集：

```
import pylab as pl
pl.plot(x,y, 'ro')
```

B样条曲线插值-举例

绘制数据点集：



B样条曲线插值-举例

创建interp1d对象f、计算插值结果：

```
xnew = np.linspace(0, 10, 11)
from scipy import interpolate
f = interpolate.interp1d(x, y, kind = kind)
ynew = f(xnew)
```

B样条曲线插值-举例

根据kind类型创建interp1d对象f、计算并绘制插值结果：

```
xnew = np.linspace(0, 10, 11)
for kind in ['nearest', 'zero', 'linear', 'quadratic']:
    #根据kind创建插值对象interp1d
    f = interpolate.interp1d(x, y, kind = kind)
    ynew = f(xnew)#计算插值结果
    pl.plot(xnew, ynew, label = str(kind))#绘制插值结果
```

B样条曲线插值-举例

```
import numpy as np
from scipy import interpolate
import pylab as pl
#创建数据点集并绘制
x = np.linspace(0, 10, 11)
y = np.sin(x)
pl.plot(x,y,'ro')
#建立插值数据点
xnew = np.linspace(0, 10, 101)
for kind in ['nearest', 'zero', 'linear', 'quadratic']:
    #根据kind创建插值对象interp1d
    f = interpolate.interp1d(x, y, kind = kind)
    ynew = f(xnew)#计算插值结果
    pl.plot(xnew, ynew, label = str(kind))
pl.legend(loc = 'lower right')
pl.show()
```

B样条曲线插值-举例

