Scipy Basics

数 据 嗨 客

SciPy

SciPy库是科学计算的核心软件包之一，它提供基于Python的NumPy扩展构建的数学算法和便利功能.

与Numpy交互

```
import numpy as np
```

创建数组

```
a = np.array([1, 2, 3])
b = np.array([[(4, 5, 6)], [(3, 2, 1), (4, 5, 6)]])
```

索引技巧

`np.mgrid[0:6, 0:6]` | 创建一个密集的网格.
`np.ogrid[0:2, 0:2]` | 创建一个开放的网格.
`np.r_[4, [0]*2, -1:1:3j]` | 按行垂直堆栈数组，array([4., 0., 0., -1., 0., 1.]).
`np.c_[np.array([1, 2]), np.array([4, 5])]` 按列堆栈数组，array([[1, 4], [2, 5]]).

形状操纵

`np.transpose(b)` | 排列数组的维度.
`b.flatten()` | 扁平化数组.
`np.hsplit(b, 2)` | 在第二个索引水平分割数组.
`np.vpslit(b, 2)` | 在第二个索引垂直分割数组.

多项式

```
from numpy import poly1d
```
`p = poly1d([3, 4, 5])` | 创建一个多项式对象，表示返回$p = 3x^2 + 4x + 5$.

向量化函数

```
def myfunc(a, b):
    if a > b:
        return a-b
    else:
        return a+b
xfunc[[2, 4], 2]
```
`xfunc = np.vectorize(myfunc)` | 向量化函数，array([4, 2]).

类型处理

`np.real()` | 返回数组元素的实数部分.
`np.imag()` | 返回数组元素的虚部.
`np.real_if_close([2.3+4e-18j]` | 如果复杂部分接近0，则返回一个真实数组，array([2.3]).

```
x=np.array[1.2, 2.3]
```
`x.astype(int)` | 数组的副本，转换为指定的类型，array([1, 2]).

其他有用功能

`np.angle(b, deg=True)` | 返回复杂参数的角度.
`g=np.linspace(0, 6, num=5)` | 以指定的时间间隔返回均匀间隔的数字，0为序列的起始值，6为序列的结束值，num为要生成的样本数量.
`np.logspace(0, 5, 8)` | 创建等比数列,参数分别为开始点10^0，结束点10^5，元素个数8.
`np.select([c<4, c>6], [c*2, c])` | 根据条件返回从选择列表中的元素中抽取的数组,当满足c<4时，执行c*2；当c>6时，执行c.

线性代数

```
from scipy import linalg as LA
```

创建矩阵

```
c = np.mat([[3, 2, 0], [1, 1, 0], [0, 5, 1]])
d = np.matrix([0, -1, 2])
```

矩阵基础

`LA.inv(c)` | 矩阵的逆.
`LA.solve(c, d)` | 求解未知线性方程组$cx = d$.
`LA.solve_circulant(c,d)` | 对x求解$cx = d$，其中c是一个循环矩阵.
`LA.det(c)` | 计算矩阵的行列式.
`LA.pinv(c)` | 计算矩阵的伪逆.
`LA.pinvh()` | 计算Hermitian矩阵的伪逆.
`LA.norm(c)` | Frobenius范数.
`LA.norm(c, 1)` | ℓ_1范数（最大列总和）.
`LA.norm(c, np.inf)` | ℓ_∞范数（最大行总和）.
`np.linalg.matrix_rank(c)` | 矩阵的秩.
`LA.eigvals(c)` | 计算特征值.

分解

`LA.lu(c)` | 计算矩阵的枢轴LU分解.
`LA.svd(c)` | 奇异值分解.
`LA.svdvals(c)` | 计算矩阵的奇异值.
`LA.qr(c)` | 计算矩阵的QR分解.
`LA.schur(c)` | 计算矩阵的Schur分解.

矩阵函数

`LA.logm(c)` | 计算矩阵对数.
`LA.sinm(c)` | 计算矩阵正弦.
`LA.tanm(c)` | 计算矩阵正切.
`LA.coshm(c)` | 计算双曲线矩阵余弦.
`LA.sqrtm(c)` | 矩阵平方根.
`LA.expm(c)` | 矩阵指数.
`LA.logm(c)` | 矩阵对数.
`LA.fractional_matrix_power(c, 0.5)` | 计算矩阵的分数幂.
`LA.solve_sylvester(O, P, Q)` | 计算Sylvester方程的解$(Ox + xP = Q)$.

特殊矩阵

`LA.hadamard(n)` | 构建一个Hadamard矩阵，且n必须是2的幂次方.
`LA.companion([1, 3, 5])` | 创建伴随矩阵.
`LA.dft(n)` | 离散傅里叶变换矩阵（n为矩阵的大小）.
`LA.helmert(5)` | 创建一个5阶Helmert矩阵.
`LA.hilbert(3)` | 创建一个3阶Hilbert矩阵.
`LA.invhilbert(n)` | 计算n阶Hilbert矩阵的逆.

稀疏矩阵

```
from scipy import sparse as sps
```

稀疏矩阵类型

`sps.bsr_matrix((3, 4))` | 块稀疏行矩阵.
`sps.coo_matrix((3, 4))` | COOrdinate稀疏矩阵.
`sps.csc_matrix((3, 4))` | 压缩稀疏列矩阵.
`sps.csr_matrix((3, 4))` | 压缩稀疏行矩阵.
`sps.dia_matrix([1, 2, 3], 1)` | 具有DIAgonal存储的稀疏矩阵.
`sps.dok_matrix((5, 5))`|基于字典键的稀疏矩阵.
`sps.lil_matrix(c)` | 基于行的链表稀疏矩阵.

函数

`sps.eye(3)` | 对角线上创建全为1的稀疏矩阵.
`sps.kron(X, Y)` | 稀疏矩阵X和Y的kronecker乘积.
`sps.kronsum(X, Y)` | 稀疏矩阵X和Y的kronecker和.
`sps.diags([1, -2, 1], [-1, 0, 1])` | 将数组放在稀疏矩阵的指定对角线上.
`sps.spdiags(c, [0, -1, 2], 4, 4)` | 将高维数组放在稀疏矩阵的指定对角线上.
`sps.tril(c)` | 以稀疏格式返回矩阵的下三角部分.
`sps.triu(c)` | 以稀疏格式返回矩阵的上三角部分.
`sps.rand(3, 4)` | 生成具有均匀分布值的给定形状和密度的稀疏矩阵.
`sps.random(4, 5)` | 生成具有随机分布值的给定形状和密度的稀疏矩阵.

保存并加载稀疏矩阵

`sps.save_npz(file)` | 将稀疏矩阵保存到文件中.
`sps.load_npz(file)` | 从文件中加载稀疏矩阵.

识别稀疏矩阵

`sps.issparse(x)` | x是否为一个稀疏矩阵类型.
`sps.isspmatrix_csc(x)` | x是否为csc matrix类型.
`sps.isspmatrix_csr(x)` | x是否为csr matrix类型.
`sps.isspmatrix_bsr(x)` | x是否为bsr matrix类型.

常用子模块

scipy.cluster	矢量量化/Kmeans
scipy.constants	物理和数学常数
scipy.fftpack	傅里叶变换
scipy.integrate	集成例程
scipy.interpolate	插值
scipy.io	数据输入和输出
scipy.linalg	线性代数例程
scipy.ndimage	n维图像包
scipy.odr	正交距离回归
scipy.optimize	积分和常微分方程求解
scipy.signal	信号处理
scipy.sparse	稀疏矩阵
scipy.spatial	空间数据结构和算法
scipy.special	信号处理
scipy.stats	统计