SciPy

点击一键运行代码→ | 获取更多速查表→ | 使用说明书 [视频 & 图文]→ bilibili Show Me AI

1. 与 NumPy 交互

初始化

```
> import numpy as np
  a = np.array([1, 2, 3])
  b = np.array([(1+5j, 2j, 3j), (4j, 5j, 6j)])
  c = np.array([[(1.5, 2, 3), (4, 5, 6)], [(3, 2, 1), (4, 5, 6)]])
```

索引技巧

```
> np.mgrid[0:5, 0:5] # 创建稠密栅格
> np.ogrid[0:2, 0:2] # 创建开放栅格
> np.r_[3, [0]*5, -1:1:10j] # 按行纵向堆叠数组
> np.c_[a, a] # 按列横向堆叠数组
```

操控形状

```
> np.transpose(b) # 转置矩阵
> b.flatten() # 拉平数组
> np.hstack((c, c)) # 按列横向堆叠数组
> np.vstack((a, b)) # 按行纵向堆叠数组
> np.hsplit(c, 2) # 在索引 2 横向分割数组
> np.vsplit(c, 2) # 在索引 2 纵向分割数组
```

常用函数

```
> np.angle(b, deg=True) # 返回复数的辐角
> g = np.linspace(0, np.pi, num=5) # 创建等差数组 ( 样本数 )
> g [3:] += np.pi
> np.unwrap(g) # 解包
> np.logspace(0, 10, 3) # 创建等差数组 ( 对数刻度 )
> np.select([c<4], [c*2]) # 根据条件返回数组列表的值

> from scipy import special
  special.factorial(a) # 因子

> import scipy
  scipy.special.comb(10, 3, exact=True) # 计算排列组合 C_10^3

> from scipy import misc
  misc.central_diff_weights(3) #NP 点中心导数的权重
  misc.derivative(myfunc, 1.0) # 查找函数在某点的第 n 个导数
```

矢量函数

```
> def myfunc(a):
      if a < 0:
              return a*2
      else:
              return a/2
  np.vectorize(myfunc)  # 矢量函数
```

类型控制

```
> np.real(c) # 返回数组元素的实部
> np.imag(c) # 返回数组元素的虚部
 # 如果复数接近 0，返回实部
> np.real_if_close(c, tol=1000)
> np.cast['f'](np.pi) # 将对象转化为数据类型
```

多项式

```
> from numpy import poly1d
  p = poly1d([3, 4, 5]) # 创建多项式对象
```

作者 | 韩信子 @ShowMeAI
设计 | 南　乔 @ShowMeAI
参考 | datacamp cheatsheet

扫码回复 "速查表"
下载最新全套资料

2. 线性代数

使用 linalg 和 sparse 模块。
注意 scipy.linalg 包含了 numpy.linalg，并扩展了其功能。

```
> from scipy import linalg, sparse
```

[2.1] 创建矩阵

```
> A = np.matrix(np.random.random((2, 2)))
> B = np.asmatrix(b)
> C = np.mat(np.random.random((10, 5)))
> D = np.mat([[3, 4], [5, 6]])
```

[2.2] 基础矩阵操作

逆矩阵

```
> A.I # 求逆矩阵
> linalg.inv(A) # 求逆矩阵
> A.T # 矩阵转置
> A.H # 共轭转置
> np.trace(A) # 计算对角线元素的和
```

范数

```
> linalg.norm(A) #Frobeniu 范数
> linalg.norm(A, 1) #L1 范数 ( 最大列汇总 )
> linalg.norm(A, np.inf) #L 范数 ( 最大列汇总 )
```

排名

```
> np.linalg.matrix_rank(C) # 矩阵排名
```

行列式

```
> linalg.det(A) # 行列式
```

求解线性问题

```
> linalg.solve(A, b)  # 求解稠密矩阵
> E = np.mat(a).T     # 求解稠密矩阵
> linalg.lstsq(D, D) # 用最小二乘法求解线性代数方程
```

广义逆

```
> linalg.pinv(C)   # 计算矩阵的伪逆 ( 最小二乘法求解器 )
> linalg.pinv2(C) # 计算矩阵的伪逆 (SVD)
```

2. 线性代数

[2.3] 创建稀疏矩阵

```
> F = np.eye(3, k=1) # 创建 2X2 单位矩阵
> G = np.mat(np.identity(2))# 同上
> C[C > 0.5] = 0

> H = sparse.csr_matrix(C) # 压缩稀疏行矩阵
> I = sparse.csc_matrix(D) # 压缩稀疏列矩阵

> J = sparse.dok_matrix(A) #DOK 矩阵
> I.todense() # 将稀疏矩阵转为全矩阵
> sparse.isspmatrix_csc(A) # 单位稀疏矩阵
```

[2.4] 稀疏矩阵操作

逆矩阵

```
> import scipy.sparse.linalg as linalg
  linalg.inv(I) # 求逆矩阵
```

范数

```
  linalg.norm(I) # 范数
```

解决线性问题

```
  linalg.spsolve(I, I) # 稀求解疏矩阵
```

[2.5] 稀疏矩阵函数

```
> sparse.linalg.expm(I) # 稀疏矩阵指数
```

[2.6] 矩阵函数

加法

```
> np.add(A, D) # 加法
```

减法

```
> np.subtract(A, D) # 减法
```

除法

```
> np.divide(A, D) # 除法
```

乘法

```
> np.multiply(D, A) # 乘法
> np.dot(A, D) # 点积
> np.vdot(A, D) # 向量点积
> np.inner(A, D) # 内积
> np.outer(A, D) # 外积
> np.tensordot(A, D) # 张量点积
> np.kron(A, D) #Kronecker 积
```

指数函数

```
> linalg.expm(A) # 矩阵指数
```

对数函数

```
> scipy.linalg.logm(A) # 矩阵对数
```

三角函数

```
> scipy.linalg.sinm(D) # 矩阵正弦
> scipy.linalg.cosm(D) # 矩阵余弦
> scipy.linalg.tanm(A) # 矩阵切线
```

双曲三角函数

```
> scipy.linalg.sinhm(D) # 双曲矩阵正弦
> scipy.linalg.coshm(D) # 双曲矩阵余弦
> scipy.linalg.tanhm(A) # 双曲矩阵切线
```

矩阵符号函数

```
> np.sign(A) # 矩阵符号函数
```

矩阵平方根

```
> scipy.linalg.sqrtm(A) # 矩阵平方根
```

任意函数

```
# 评估矩阵函数
> scipy.linalg.funm(A, lambda x: x*x)
```

[2.7] 矩阵分解

特征值与特征向量

```
# 求解方阵的普通或广义特征值问题
> la, v = scipy.linalg.eig(A)
> l1, l2 = la # 解包特征值
> v[:, 0] # 第一个特征值
> v[:, 1] # 第二个特征值
> scipy.linalg.eigvals(A) # 解包特征值
```

奇异值分解 (SVD)

```
> U, s, Vh = scipy.linalg.svd(B)
> M, N = B.shape
# 在 SVD 中构建 Sigma 矩阵
> Sig = scipy.linalg.diagsvd(s, M, N)
```

LU 分解

```
> P, L, U = scipy.linalg.lu(C) #LU 分解
```

解构稀疏矩阵

```
# 特征值与特征向量
> la, v = sparse.linalg.eigs(F, 1)
# 奇异值分解 (SVD)
> sparse.linalg.svds(H, 2)
```

SciPy 是著名的 python 开源科学计算库。SciPy 构建于 NumPy 之上进行科学计算，统计分析。

SciPy 提供了许多科学计算的库函数，如线性代数、微分方程、信号处理、图像处理、系数矩阵计算等。

SciPy

调用帮助

help 函数

```
> help(scipy.linalg.diagsvd)
> np.info(np.matrix)
```

ShowMeAI