1、Numpy是什么

很简单，Numpy是Python的一个科学计算的库，提供了矩阵运算的功能，其一般与Scipy、matplotlib一起使用。其实，list已经提供了类似于矩阵的表示形式，不过numpy为我们提供了更多的函数。如果接触过matlab、scilab，那么numpy很好入手。 在以下的代码示例中，总是先导入了numpy：

复制代码 代码如下:

>>> import numpy as np  
>>> print np.version.version  
1.6.2

2、多维数组

多维数组的类型是：numpy.ndarray。

使用numpy.array方法

以list或tuple变量为参数产生一维数组：

复制代码 代码如下:

>>> print np.array([1,2,3,4])  
[1 2 3 4]  
>>> print np.array((1.2,2,3,4))  
[ 1.2  2.   3.   4. ]  
>>> print type(np.array((1.2,2,3,4)))  
<type 'numpy.ndarray'>

以list或tuple变量为元素产生二维数组：

复制代码 代码如下:

>>> print np.array([[1,2],[3,4]])  
[[1 2]  
 [3 4]]

生成数组的时候，可以指定数据类型，例如numpy.int32, numpy.int16, and numpy.float64等：

复制代码 代码如下:

>>> print np.array((1.2,2,3,4), dtype=np.int32)  
[1 2 3 4]

使用numpy.arange方法

复制代码 代码如下:

>>> print np.arange(15)  
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14]  
>>> print type(np.arange(15))  
<type 'numpy.ndarray'>  
>>> print np.arange(15).reshape(3,5)  
[[ 0  1  2  3  4]  
 [ 5  6  7  8  9]  
 [10 11 12 13 14]]  
>>> print type(np.arange(15).reshape(3,5))  
<type 'numpy.ndarray'>

使用numpy.linspace方法

例如，在从1到3中产生9个数：

复制代码 代码如下:

>>> print np.linspace(1,3,9)  
[ 1.    1.25  1.5   1.75  2.    2.25  2.5   2.75  3.  ]

使用numpy.zeros，numpy.ones，numpy.eye等方法可以构造特定的矩阵

例如：

复制代码 代码如下:

>>> print np.zeros((3,4))  
[[ 0.  0.  0.  0.]  
 [ 0.  0.  0.  0.]  
 [ 0.  0.  0.  0.]]  
>>> print np.ones((3,4))  
[[ 1.  1.  1.  1.]  
 [ 1.  1.  1.  1.]  
 [ 1.  1.  1.  1.]]  
>>> print np.eye(3)  
[[ 1.  0.  0.]  
 [ 0.  1.  0.]  
 [ 0.  0.  1.]]

创建一个三维数组：

复制代码 代码如下:

>>> print np.zeros((2,2,2))  
[[[ 0.  0.]  
  [ 0.  0.]]

 [[ 0.  0.]  
  [ 0.  0.]]]

获取数组的属性：

复制代码 代码如下:

>>> a = np.zeros((2,2,2))  
>>> print a.ndim   #数组的维数  
3  
>>> print a.shape  #数组每一维的大小  
(2, 2, 2)  
>>> print a.size   #数组的元素数  
8  
>>> print a.dtype  #元素类型  
float64  
>>> print a.itemsize  #每个元素所占的字节数  
8

数组索引，切片，赋值

示例：

复制代码 代码如下:

>>> a = np.array( [[2,3,4],[5,6,7]] )  
>>> print a  
[[2 3 4]  
 [5 6 7]]  
>>> print a[1,2]  
7  
>>> print a[1,:]  
[5 6 7]  
>>> print a[1,1:2]  
[6]  
>>> a[1,:] = [8,9,10]  
>>> print a  
[[ 2  3  4]  
 [ 8  9 10]]

使用for操作元素

复制代码 代码如下:

>>> for x in np.linspace(1,3,3):  
...     print x  
...  
1.0  
2.0  
3.0

基本的数组运算

先构造数组a、b：

复制代码 代码如下:

>>> a = np.ones((2,2))  
>>> b = np.eye(2)  
>>> print a  
[[ 1.  1.]  
 [ 1.  1.]]  
>>> print b  
[[ 1.  0.]  
 [ 0.  1.]]

数组的加减乘除：

复制代码 代码如下:

>>> print a > 2  
[[False False]  
 [False False]]  
>>> print a+b  
[[ 2.  1.]  
 [ 1.  2.]]  
>>> print a-b  
[[ 0.  1.]  
 [ 1.  0.]]  
>>> print b\*2  
[[ 2.  0.]  
 [ 0.  2.]]  
>>> print (a\*2)\*(b\*2)  
[[ 4.  0.]  
 [ 0.  4.]]  
>>> print b/(a\*2)  
[[ 0.5  0. ]  
 [ 0.   0.5]]  
>>> print (a\*2)\*\*4  
[[ 16.  16.]  
 [ 16.  16.]]

 使用数组对象自带的方法：

复制代码 代码如下:

>>> a.sum()  
4.0  
>>> a.sum(axis=0)   #计算每一列（二维数组中类似于矩阵的列）的和  
array([ 2.,  2.])  
>>> a.min()  
1.0  
>>> a.max()  
1.0

使用numpy下的方法：

复制代码 代码如下:

>>> np.sin(a)  
array([[ 0.84147098,  0.84147098],  
       [ 0.84147098,  0.84147098]])  
>>> np.max(a)  
1.0  
>>> np.floor(a)  
array([[ 1.,  1.],  
       [ 1.,  1.]])  
>>> np.exp(a)  
array([[ 2.71828183,  2.71828183],  
       [ 2.71828183,  2.71828183]])  
>>> np.dot(a,a)   ##矩阵乘法  
array([[ 2.,  2.],  
       [ 2.,  2.]])

合并数组

使用numpy下的vstack和hstack函数：

复制代码 代码如下:

>>> a = np.ones((2,2))  
>>> b = np.eye(2)  
>>> print np.vstack((a,b))  
[[ 1.  1.]  
 [ 1.  1.]  
 [ 1.  0.]  
 [ 0.  1.]]  
>>> print np.hstack((a,b))  
[[ 1.  1.  1.  0.]  
 [ 1.  1.  0.  1.]]

看一下这两个函数有没有涉及到浅拷贝这种问题：

复制代码 代码如下:

>>> c = np.hstack((a,b))  
>>> print c  
[[ 1.  1.  1.  0.]  
 [ 1.  1.  0.  1.]]  
>>> a[1,1] = 5  
>>> b[1,1] = 5  
>>> print c  
[[ 1.  1.  1.  0.]  
 [ 1.  1.  0.  1.]]

可以看到，a、b中元素的改变并未影响c。

深拷贝数组

数组对象自带了浅拷贝和深拷贝的方法，但是一般用深拷贝多一些：

复制代码 代码如下:

>>> a = np.ones((2,2))  
>>> b = a  
>>> b is a  
True  
>>> c = a.copy()  #深拷贝  
>>> c is a  
False

基本的矩阵运算

转置：

复制代码 代码如下:

>>> a = np.array([[1,0],[2,3]])  
>>> print a  
[[1 0]  
 [2 3]]  
>>> print a.transpose()  
[[1 2]  
 [0 3]]

迹：

复制代码 代码如下:

>>> print np.trace(a)  
4

numpy.linalg模块中有很多关于矩阵运算的方法：

复制代码 代码如下:

>>> import numpy.linalg as nplg

特征值、特征向量：

复制代码 代码如下:

>>> print nplg.eig(a)  
(array([ 3.,  1.]), array([[ 0.        ,  0.70710678],  
       [ 1.        , -0.70710678]]))