

## 3.10 矩阵与线性代数运算¶

### 问题¶

你需要执行矩阵和线性代数运算，比如矩阵乘法、寻找行列式、求解线性方程组等等。

### 解决方案¶

NumPy 库有一个矩阵对象可以用来解决这个问题。

矩阵类似于3.9小节中数组对象，但是遵循线性代数的计算规则。下面的一个例子展示了矩阵的一些基本特性：

```
>>> import numpy as np
>>> m = np.matrix([[1,-2,3],[0,4,5],[7,8,-9]])
>>> m
matrix([[ 1, -2,  3],
        [ 0,  4,  5],
        [ 7,  8, -9]])

>>> # Return transpose
>>> m.T
matrix([[ 1,  0,  7],
        [-2,  4,  8],
        [ 3,  5, -9]])

>>> # Return inverse
>>> m.I
matrix([[ 0.33043478, -0.02608696,  0.09565217],
        [-0.15217391,  0.13043478,  0.02173913],
        [ 0.12173913,  0.09565217, -0.0173913 ]])

>>> # Create a vector and multiply
>>> v = np.matrix([[2],[3],[4]])
>>> v
matrix([[2],
        [3],
        [4]])
>>> m * v
matrix([[ 8],
        [32],
        [ 2]])

>>>
```

可以在 `numpy.linalg` 子包中找到更多的操作函数，比如：

```
>>> import numpy.linalg

>>> # Determinant
>>> numpy.linalg.det(m)
-229.99999999999983

>>> # Eigenvalues
>>> numpy.linalg.eigvals(m)
array([-13.11474312,  2.75956154,  6.35518158])

>>> # Solve for x in mx = v
>>> x = numpy.linalg.solve(m, v)
>>> x
matrix([[ 0.96521739],
        [ 0.17391304],
        [ 0.46086957]])
>>> m * x
matrix([[ 2.],
        [ 3.],
        [ 4.]])

>>> v
```

```
matrix([[2],  
        [3],  
        [4]])  
>>>
```

## 讨论¶

很显然线性代数是个非常大的主题，已经超出了本书能讨论的范围。但是，如果你需要操作数组和向量的话，NumPy 是一个不错的入口点。可以访问 NumPy 官网 <http://www.numpy.org> 获取更多信息。