Numpy Maths

August 6, 2020

This tutorial gives a quick overview of the Numpy Maths

Reading an array from a file

Numpy Maths

Combining arrays

Basic algebraic operations using numpy arrays

Solving linear equations

Matix inversions

Calculating eigen vectors

1 Reading an array from a file

1.0.1 narray.txt

[2,4,6,7

1,2,3,5

3,2,7,8

1,5,6,4]

```
[1]: #load txt file using loadtxt function
     import numpy as np
     f = np.loadtxt("narray.txt", delimiter=',')
```

```
[2]: np.load?
```

```
[3]: f
```

```
[3]: array([[2., 4., 6., 7.],
            [1., 2., 3., 5.],
            [3., 2., 7., 8.],
            [1., 5., 6., 4.]])
```

```
[4]: #data type  conversation
     f = np.loadtxt("narray.txt", delimiter=',', dtype = "int")
```

```
[5]: f
```

```
[5]: array([[2, 4, 6, 7],
            [1, 2, 3, 5],
            [3, 2, 7, 8],
            [1, 5, 6, 4]])
```

```
[6]: #datatype
     f.dtype
```

```
[6]: dtype('int32')
```

2 NUMPY Maths

2.1 Adding, subtracting, multiplying, transposing arryas

```
[7]: # Generating 5*5 size array using random function for intger datatype
     import numpy as np
     x = np.random.randint( 100, size = (5,5) )
     y = np.random.randint( 100, size = (5,5) )
```

```
[8]: x
```

```
[8]: array([[93, 28, 90, 85, 52],
            [34, 97, 87, 28, 45],
            [72, 41, 93, 46, 53],
            [53, 13, 67, 79, 48],
            [83, 16, 88, 52, 50]])
```

```
[9]: y
```

```
[9]: array([[23, 45, 50, 53, 88],
            [38, 98, 29, 44, 51],
            [19, 88, 47, 30, 19],
            [38, 66, 88, 58, 51],
            [97, 44, 60, 51, 26]])
```

```
[10]: ## Add two matrices x + y or np.add( x, y )
      x + y
```

```
[10]: array([[116,  73, 140, 138, 140],
             [ 72, 195, 116,  72,  96],
             [ 91, 129, 140,  76,  72],
             [ 91,  79, 155, 137,  99],
             [180,  60, 148, 103,  76]])
```

```
[11]: np.add( x, y )
```

```
[11]: array([[116,  73, 140, 138, 140],
             [ 72, 195, 116,  72,  96],
             [ 91, 129, 140,  76,  72],
             [ 91,  79, 155, 137,  99],
             [180,  60, 148, 103,  76]])
```

```
[12]: # np.substract( x, y )
      x - y
```

```
[12]: array([[ 70, -17,  40,  32, -36],
             [ -4,  -1,  58, -16,  -6],
             [ 53, -47,  46,  16,  34],
             [ 15, -53, -21,  21,  -3],
             [-14, -28,  28,   1,  24]])
```

```
[13]: # np.multiply( x, y )
      x * y
```

```
[13]: array([[2139, 1260, 4500, 4505, 4576],
             [1292, 9506, 2523, 1232, 2295],
             [1368, 3608, 4371, 1380, 1007],
             [2014,  858, 5896, 4582, 2448],
             [8051,  704, 5280, 2652, 1300]])
```

```
[14]: # Matrix Transpose
      x.T
```

```
[14]: array([[93, 34, 72, 53, 83],
             [28, 97, 41, 13, 16],
             [90, 87, 93, 67, 88],
             [85, 28, 46, 79, 52],
             [52, 45, 53, 48, 50]])
```

3 Calculating column sums and row sums

```
[15]: x = np.random.randint( 10, size = (4,4) )
```

```
[16]: x
```

```
[16]: array([[2, 3, 3, 7],
             [4, 4, 5, 7],
             [6, 2, 6, 5],
             [3, 4, 5, 0]])
```

```
[17]: print (x.sum(), x.mean(), x.std())
```

```
66 4.125 1.8666480653835098
```

```
[18]: np.sum(x, axis=0)
```

```
[18]: array([15, 13, 19, 19])
```

```
[19]: x+[10,15,20,30]
```

```
[19]: array([[12, 18, 23, 37],
             [14, 19, 25, 37],
             [16, 17, 26, 35],
             [13, 19, 25, 30]])
```

```
[20]: l=np.array([[10,20], [20,30]])
```

```
[21]: l+10
```

```
[21]: array([[20, 30],
             [30, 40]])
```

```
[22]: x
```

```
[22]: array([[2, 3, 3, 7],
             [4, 4, 5, 7],
             [6, 2, 6, 5],
             [3, 4, 5, 0]])
```

```
[23]: np.std(x)
```

```
[23]: 1.8666480653835098
```

```
[24]: np.sum( x, axis = 0 )
```

```
[24]: array([15, 13, 19, 19])
```

```
[25]: np.sum( x, axis = 1 )
```

```
[25]: array([15, 20, 19, 12])
```

```
[26]: np.mean( x, axis = 0 )
```

```
[26]: array([3.75, 3.25, 4.75, 4.75])
```

```
[27]: np.mean( x, axis = 1 )
```

```
[27]: array([3.75, 5.  , 4.75, 3.  ])
```

4 Combining / Concatenating Arrays

```
[28]: import numpy as np
      x = np.random.randint( 10, size = (2,2) )
      y = np.random.randint( 100, size = (2,2) )
```

```
[29]: x
```

```
[29]: array([[7, 0],
             [4, 1]])
```

```
[30]: y
```

```
[30]: array([[53, 88],
             [28,  7]])
```

```
[31]: np.vstack( [x, y])
```

```
[31]: array([[ 7,  0],
             [ 4,  1],
             [53, 88],
             [28,  7]])
```

```
[32]: np.hstack([x, y])
```

```
[32]: array([[ 7,  0, 53, 88],
             [ 4,  1, 28,  7]])
```

```
[33]: import numpy as np
      np.r_[x,y]
```

```
[33]: array([[ 7,  0],
             [ 4,  1],
             [53, 88],
             [28,  7]])
```

```
[34]: np.c_[x,y]
```

```
[34]: array([[ 7,  0, 53, 88],
             [ 4,  1, 28,  7]])
```

5 Linear Algebra.. Advanced Matrix Operation

[35]:
```
from numpy import linalg
import numpy as np
```

6 Solving a set of linear equations

2x + 2y+3z = 5 3x + y +4z = 7 4x + 3y =10

[36]:
```
a = np.array([[2,2,3], [3,1,4],[4,3,0]])
b = np.array([5,7,10])
x = np.linalg.solve(a, b)
x
```

[36]:
```
array([ 2.30434783,  0.26086957, -0.04347826])
```

7 Matrix Inversion

[37]:
```
a = np.array([[1, 2], [3, 4]])
np.linalg.inv( a )
```

[37]:
```
array([[-2. ,  1. ],
       [ 1.5, -0.5]])
```

8 Calculating an eigen value and vector for a matrix

[38]:
```
m1 = np.diag((1, 2, 3))
m1
```

[38]:
```
array([[1, 0, 0],
       [0, 2, 0],
       [0, 0, 3]])
```

[39]:
```
eigval, eigvec = np.linalg.eig( m1 )
```

[40]:
```
eigval
```

[40]:
```
array([1., 2., 3.])
```

[41]:
```
eigvec
```

[41]:
```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```