Introduction to Numpy

October 29, 2019

1 Introduction To Numpy

```
In [1]: import numpy as NP
```

Creating Array Using Numpy

```
In [2]: NP.array([1,2,3,4])

Out[2]: array([1, 2, 3, 4])

In [3]: NP.array([[255 ,5,5],[5,2,7,8],[85,25,14,45]])

Out[3]: array([list([255, 5, 5]), list([5, 2, 7, 8]), list([85, 25, 14, 45])],
              dtype=object)

In [4]: NP.array([[255 ,5,6,5],[5,2,7,8],[85,25,14,45]])

Out[4]: array([[255,   5,   6,   5],
               [  5,   2,   7,   8],
               [ 85,  25,  14,  45]])

In [5]: List = [1,5,6,9,8,7]

In [6]: NP.array([List,List,List,List])

Out[6]: array([[1, 5, 6, 9, 8, 7],
               [1, 5, 6, 9, 8, 7],
               [1, 5, 6, 9, 8, 7],
               [1, 5, 6, 9, 8, 7]])
```

Creating Diagonal Array

```
In [7]: NP.eye(5)

Out[7]: array([[1., 0., 0., 0., 0.],
               [0., 1., 0., 0., 0.],
               [0., 0., 1., 0., 0.],
               [0., 0., 0., 1., 0.],
               [0., 0., 0., 0., 1.]])
```

```
In [8]: 10 * NP.eye(5)

Out[8]: array([[10.,  0.,  0.,  0.,  0.],
               [ 0., 10.,  0.,  0.,  0.],
               [ 0.,  0., 10.,  0.,  0.],
               [ 0.,  0.,  0., 10.,  0.],
               [ 0.,  0.,  0.,  0., 10.]])
```

Creating Zeroes Array

```
In [9]: 0*NP.eye(5)

Out[9]: array([[0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0.]])

In [10]: NP.zeros(5)

Out[10]: array([0., 0., 0., 0., 0.])

In [11]: NP.zeros(shape= (5,5))

Out[11]: array([[0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0.]])
```

Creating Ones Array

```
In [12]: NP.ones(5)

Out[12]: array([1., 1., 1., 1., 1.])

In [13]: NP.ones(shape = (4,4))

Out[13]: array([[1., 1., 1., 1.],
                [1., 1., 1., 1.],
                [1., 1., 1., 1.],
                [1., 1., 1., 1.]])

In [14]: 8 * NP.ones(shape = (5,6))

Out[14]: array([[8., 8., 8., 8., 8., 8.],
                [8., 8., 8., 8., 8., 8.],
                [8., 8., 8., 8., 8., 8.],
                [8., 8., 8., 8., 8., 8.],
                [8., 8., 8., 8., 8., 8.]])
```

Creating Range list using Numpy

```
In [15]: NP.arange(10)

Out[15]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

In [16]: NP.arange(5,50)

Out[16]: array([ 5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21,
                22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38,
                39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49])

In [17]: NP.arange(10,250,2)

Out[17]: array([ 10,  12,  14,  16,  18,  20,  22,  24,  26,  28,  30,  32,  34,
                 36,  38,  40,  42,  44,  46,  48,  50,  52,  54,  56,  58,  60,
                 62,  64,  66,  68,  70,  72,  74,  76,  78,  80,  82,  84,  86,
                 88,  90,  92,  94,  96,  98, 100, 102, 104, 106, 108, 110, 112,
                114, 116, 118, 120, 122, 124, 126, 128, 130, 132, 134, 136, 138,
                140, 142, 144, 146, 148, 150, 152, 154, 156, 158, 160, 162, 164,
                166, 168, 170, 172, 174, 176, 178, 180, 182, 184, 186, 188, 190,
                192, 194, 196, 198, 200, 202, 204, 206, 208, 210, 212, 214, 216,
                218, 220, 222, 224, 226, 228, 230, 232, 234, 236, 238, 240, 242,
                244, 246, 248])

In [18]: 10*NP.arange(250,10,-3)

Out[18]: array([2500, 2470, 2440, 2410, 2380, 2350, 2320, 2290, 2260, 2230, 2200,
                2170, 2140, 2110, 2080, 2050, 2020, 1990, 1960, 1930, 1900, 1870,
                1840, 1810, 1780, 1750, 1720, 1690, 1660, 1630, 1600, 1570, 1540,
                1510, 1480, 1450, 1420, 1390, 1360, 1330, 1300, 1270, 1240, 1210,
                1180, 1150, 1120, 1090, 1060, 1030, 1000,  970,  940,  910,  880,
                 850,  820,  790,  760,  730,  700,  670,  640,  610,  580,  550,
                 520,  490,  460,  430,  400,  370,  340,  310,  280,  250,  220,
                 190,  160,  130])
```

Range of Numbers with Equal Partition

```
In [19]: NP.linspace(1,100,7)

Out[19]: array([  1. ,  17.5,  34. ,  50.5,  67. ,  83.5, 100. ])

In [20]: NP.linspace(0,2500 , 25)

Out[20]: array([   0.        ,  104.16666667,  208.33333333,  312.5       ,
                 416.66666667,  520.83333333,  625.        ,  729.16666667,
                 833.33333333,  937.5       , 1041.66666667, 1145.83333333,
                1250.        , 1354.16666667, 1458.33333333, 1562.5       ,
                1666.66666667, 1770.83333333, 1875.        , 1979.16666667,
                2083.33333333, 2187.5       , 2291.66666667, 2395.83333333,
                2500.        ])
```

```
In [21]: NP.linspace(90,2500,250,retstep=True)

Out[21]: (array([  90.        ,   99.67871486,  109.35742972,  119.03614458,
                  128.71485944,  138.3935743 ,  148.07228916,  157.75100402,
                  167.42971888,  177.10843373,  186.78714859,  196.46586345,
                  206.14457831,  215.82329317,  225.50200803,  235.18072289,
                  244.85943775,  254.53815261,  264.21686747,  273.89558233,
                  283.57429719,  293.25301205,  302.93172691,  312.61044177,
                  322.28915663,  331.96787149,  341.64658635,  351.3253012 ,
                  361.00401606,  370.68273092,  380.36144578,  390.04016064,
                  399.7188755 ,  409.39759036,  419.07630522,  428.75502008,
                  438.43373494,  448.1124498 ,  457.79116466,  467.46987952,
                  477.14859438,  486.82730924,  496.5060241 ,  506.18473896,
                  515.86345382,  525.54216867,  535.22088353,  544.89959839,
                  554.57831325,  564.25702811,  573.93574297,  583.61445783,
                  593.29317269,  602.97188755,  612.65060241,  622.32931727,
                  632.00803213,  641.68674699,  651.36546185,  661.04417671,
                  670.72289157,  680.40160643,  690.08032129,  699.75903614,
                  709.437751  ,  719.11646586,  728.79518072,  738.47389558,
                  748.15261044,  757.8313253 ,  767.51004016,  777.18875502,
                  786.86746988,  796.54618474,  806.2248996 ,  815.90361446,
                  825.58232932,  835.26104418,  844.93975904,  854.6184739 ,
                  864.29718876,  873.97590361,  883.65461847,  893.33333333,
                  903.01204819,  912.69076305,  922.36947791,  932.04819277,
                  941.72690763,  951.40562249,  961.08433735,  970.76305221,
                  980.44176707,  990.12048193,  999.79919679, 1009.47791165,
                 1019.15662651, 1028.83534137, 1038.51405622, 1048.19277108,
                 1057.87148594, 1067.5502008 , 1077.22891566, 1086.90763052,
                 1096.58634538, 1106.26506024, 1115.9437751 , 1125.62248996,
                 1135.30120482, 1144.97991968, 1154.65863454, 1164.3373494 ,
                 1174.01606426, 1183.69477912, 1193.37349398, 1203.05220884,
                 1212.73092369, 1222.40963855, 1232.08835341, 1241.76706827,
                 1251.44578313, 1261.12449799, 1270.80321285, 1280.48192771,
                 1290.16064257, 1299.83935743, 1309.51807229, 1319.19678715,
                 1328.87550201, 1338.55421687, 1348.23293173, 1357.91164659,
                 1367.59036145, 1377.26907631, 1386.94779116, 1396.62650602,
                 1406.30522088, 1415.98393574, 1425.6626506 , 1435.34136546,
                 1445.02008032, 1454.69879518, 1464.37751004, 1474.0562249 ,
                 1483.73493976, 1493.41365462, 1503.09236948, 1512.77108434,
                 1522.4497992 , 1532.12851406, 1541.80722892, 1551.48594378,
                 1561.16465863, 1570.84337349, 1580.52208835, 1590.20080321,
                 1599.87951807, 1609.55823293, 1619.23694779, 1628.91566265,
                 1638.59437751, 1648.27309237, 1657.95180723, 1667.63052209,
                 1677.30923695, 1686.98795181, 1696.66666667, 1706.34538153,
                 1716.02409639, 1725.70281124, 1735.3815261 , 1745.06024096,
                 1754.73895582, 1764.41767068, 1774.09638554, 1783.7751004 ,
                 1793.45381526, 1803.13253012, 1812.81124498, 1822.48995984,
                 1832.1686747 , 1841.84738956, 1851.52610442, 1861.20481928,
```

```
                1870.88353414, 1880.562249  , 1890.24096386, 1899.91967871,
                1909.59839357, 1919.27710843, 1928.95582329, 1938.63453815,
                1948.31325301, 1957.99196787, 1967.67068273, 1977.34939759,
                1987.02811245, 1996.70682731, 2006.38554217, 2016.06425703,
                2025.74297189, 2035.42168675, 2045.10040161, 2054.77911647,
                2064.45783133, 2074.13654618, 2083.81526104, 2093.4939759 ,
                2103.17269076, 2112.85140562, 2122.53012048, 2132.20883534,
                2141.8875502 , 2151.56626506, 2161.24497992, 2170.92369478,
                2180.60240964, 2190.2811245 , 2199.95983936, 2209.63855422,
                2219.31726908, 2228.99598394, 2238.6746988 , 2248.35341365,
                2258.03212851, 2267.71084337, 2277.38955823, 2287.06827309,
                2296.74698795, 2306.42570281, 2316.10441767, 2325.78313253,
                2335.46184739, 2345.14056225, 2354.81927711, 2364.49799197,
                2374.17670683, 2383.85542169, 2393.53413655, 2403.21285141,
                2412.89156627, 2422.57028112, 2432.24899598, 2441.92771084,
                2451.6064257 , 2461.28514056, 2470.96385542, 2480.64257028,
                2490.32128514, 2500.        ]), 9.67871485943775)

In [22]: NP.linspace(1 , 20,3 , retstep=True ,endpoint=True )

Out[22]: (array([ 1. , 10.5, 20. ]), 9.5)

In [23]: NP.linspace(1 , 20,3 , retstep=True ,endpoint=False )

Out[23]: (array([ 1.        ,  7.33333333, 13.66666667]), 6.333333333333333)
```

Creating Random Array

```
In [24]: NP.random.rand(3,5)

Out[24]: array([[0.78028695, 0.13041731, 0.46080878, 0.73569131, 0.04772695],
                [0.06454101, 0.04236386, 0.88239712, 0.77695616, 0.99115245],
                [0.63353497, 0.93516847, 0.7783902 , 0.39388924, 0.61463196]])

In [26]: NP.random.randint(5,17 , size=(3,4))

Out[26]: array([[16, 15, 16, 15],
                [12,  9, 14,  8],
                [ 9, 15,  8, 15]])

In [27]: NP.random.randint(1,100 , 10)

Out[27]: array([91, 49,  6,  7, 33, 33, 38, 82, 28, 59])

In [31]: NP.random.rand(4,4 ,2 )

Out[31]: array([[[0.53016374, 0.6789442 ],
                 [0.65236562, 0.13754368],
                 [0.19514961, 0.47065915],
                 [0.72593327, 0.96946699]],
```

```
                [[0.93396503, 0.09617575],
                 [0.7397838 , 0.12721591],
                 [0.08189183, 0.67230237],
                 [0.95418004, 0.33185368]],

                [[0.1435755 , 0.66982972],
                 [0.96265651, 0.25226944],
                 [0.92749833, 0.84659141],
                 [0.50166716, 0.87614925]],

                [[0.82386164, 0.13232022],
                 [0.20442464, 0.82061626],
                 [0.07253223, 0.19667705],
                 [0.93588643, 0.26798676]]])

In [33]: NP.random.randn(2,3)

Out[33]: array([[-1.22131238, -0.59222238, -0.26573917],
                [ 0.40537597, -0.78398817, -1.10463575]])
```

Physical Overview of an Array Created

```
In [34]: TEST = NP.random.rand(4,4 ,2 )

In [35]: type(TEST)

Out[35]: numpy.ndarray

In [36]: TEST.size

Out[36]: 32

In [38]: TEST.shape

Out[38]: (4, 4, 2)

In [39]: TEST

Out[39]: array([[[0.58300573, 0.60311491],
                 [0.12446456, 0.04814715],
                 [0.99191032, 0.69468701],
                 [0.88297106, 0.58582532]],

                [[0.92112222, 0.98030216],
                 [0.80730714, 0.87650212],
                 [0.99428116, 0.35498873],
                 [0.94373527, 0.78608869]],

                [[0.48696104, 0.98057308],
```

```
                 [0.96402949, 0.85150286],
                 [0.04845282, 0.80777932],
                 [0.46707627, 0.40894632]],

                [[0.52273022, 0.00886216],
                 [0.45648722, 0.19516929],
                 [0.20526702, 0.70667488],
                 [0.69015401, 0.50074472]]])
```

```
In [41]: TEST.reshape(4,2,4)
```

```
Out[41]: array([[[0.58300573, 0.60311491, 0.12446456, 0.04814715],
                 [0.99191032, 0.69468701, 0.88297106, 0.58582532]],

                [[0.92112222, 0.98030216, 0.80730714, 0.87650212],
                 [0.99428116, 0.35498873, 0.94373527, 0.78608869]],

                [[0.48696104, 0.98057308, 0.96402949, 0.85150286],
                 [0.04845282, 0.80777932, 0.46707627, 0.40894632]],

                [[0.52273022, 0.00886216, 0.45648722, 0.19516929],
                 [0.20526702, 0.70667488, 0.69015401, 0.50074472]]])
```

```
In [45]: TEST.dtype
```

```
Out[45]: dtype('float64')
```

```
In [ ]:
```