NumPy Lesson - May 31, 2012

NumPy Docs: http://docs.scipy.org/doc/numpy/reference/

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

1. Building Arrays

From Other Sequences

dtypes: http://docs.scipy.org/doc/numpy/reference/arrays.scalars.html#arrays-scalars-built-in

```
In [8]: a = np.array([1, 2, 3, 10])
```

```
In [9]: print a

[ 1  2  3 10]
```

```
In [15]: b = np.array([[1, 2, 3, 4], [5, 6, 7, 10.0]], dtype=np.float32)
```

```
In [16]: b

Out[16]: array([[  1.,   2.,   3.,   4.],
                [  5.,   6.,   7.,  10.]], dtype=float32)
```

```
In [17]: b.dtype

Out[17]: dtype('float32')
```

NumPy Generation Functions

http://docs.scipy.org/doc/numpy/reference/routines.array-creation.html

```
In [19]: z = np.zeros((3, 3),dtype=np.int32)
         print z

[[0 0 0]
 [0 0 0]
 [0 0 0]]
```

```
In [21]: o = np.ones((3, 3))
         print o

[[ 1.  1.  1.]
 [ 1.  1.  1.]
 [ 1.  1.  1.]]
```

```
In [22]: a = np.arange(1, 2, 0.01)
         print a

[ 1.    1.01  1.02  1.03  1.04  1.05  1.06  1.07  1.08  1.09  1.1   1.11
  1.12  1.13  1.14  1.15  1.16  1.17  1.18  1.19  1.2   1.21  1.22  1.23
```

```
 1.24  1.25  1.26  1.27  1.28  1.29  1.3   1.31  1.32  1.33  1.34  1.35
 1.36  1.37  1.38  1.39  1.4   1.41  1.42  1.43  1.44  1.45  1.46  1.47
 1.48  1.49  1.5   1.51  1.52  1.53  1.54  1.55  1.56  1.57  1.58  1.59
 1.6   1.61  1.62  1.63  1.64  1.65  1.66  1.67  1.68  1.69  1.7   1.71
 1.72  1.73  1.74  1.75  1.76  1.77  1.78  1.79  1.8   1.81  1.82  1.83
 1.84  1.85  1.86  1.87  1.88  1.89  1.9   1.91  1.92  1.93  1.94  1.95
 1.96  1.97  1.98  1.99]
```

In [25]:
```
np.linspace(1, 2, num=50, endpoint=False)
```

Out[25]:
```
array([ 1.  ,  1.02,  1.04,  1.06,  1.08,  1.1 ,  1.12,  1.14,  1.16,
        1.18,  1.2 ,  1.22,  1.24,  1.26,  1.28,  1.3 ,  1.32,  1.34,
        1.36,  1.38,  1.4 ,  1.42,  1.44,  1.46,  1.48,  1.5 ,  1.52,
        1.54,  1.56,  1.58,  1.6 ,  1.62,  1.64,  1.66,  1.68,  1.7 ,
        1.72,  1.74,  1.76,  1.78,  1.8 ,  1.82,  1.84,  1.86,  1.88,
        1.9 ,  1.92,  1.94,  1.96,  1.98])
```

2. Indexing Arrays

http://docs.scipy.org/doc/numpy/reference/arrays.indexing.html

In [26]:
```
a = np.arange(10)
```

In [27]:
```
a
```

Out[27]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [29]:
```
a[5:8]
```

Out[29]:
```
array([5, 6, 7])
```

In [30]:
```
a = np.arange(10).reshape((2, 5))
print a
```

```
[[0 1 2 3 4]
 [5 6 7 8 9]]
```

In [33]:
```
a[1, :]
```

Out[33]:
```
array([5, 6, 7, 8, 9])
```

In [36]:
```
a[(a < 3) & (a > 1)]
```

Out[36]:
```
array([2])
```

In [38]:
```
b = np.arange(10, 20).reshape((2, 5))
print b
```

```
[[10 11 12 13 14]
 [15 16 17 18 19]]
```

In [40]:
```
b[a > 5]
```

Out[40]:
```
array([16, 17, 18, 19])
```

In [41]:
```
a > 5
```

```
Out[41]: array([[False, False, False, False, False],
                [False,  True,  True,  True,  True]], dtype=bool)
```

```
In [42]: a[a > 5] *= 2
         print a
```

```
[[ 0  1  2  3  4]
 [ 5 12 14 16 18]]
```

```
In [43]: np.where(a > 5)
```

```
Out[43]: (array([1, 1, 1, 1]), array([1, 2, 3, 4]))
```

```
In [44]: b[np.where(a > 5)]
```

```
Out[44]: array([16, 17, 18, 19])
```

3. Array Math

```
In [46]: a = np.arange(10)
         a = a * 2
         print a
```

```
[ 0  2  4  6  8 10 12 14 16 18]
```

```
In [47]: np.arange(10) * np.arange(10, 20)
```

```
Out[47]: array([  0,  11,  24,  39,  56,  75,  96, 119, 144, 171])
```

```
In [107]: a = np.arange(8).reshape((4, 2))
```

```
In [108]: a
```

```
Out[108]: array([[0, 1],
                 [2, 3],
                 [4, 5],
                 [6, 7]])
```

```
In [109]: a * np.array([2, 3])
```

```
Out[109]: array([[ 0,  3],
                 [ 4,  9],
                 [ 8, 15],
                 [12, 21]])
```

```
In [52]: a * np.array([[2], [3], [5], [6]])
```

```
Out[52]: array([[ 0,  2],
                [ 6,  9],
                [20, 25],
                [36, 42]])
```

```
In [53]: a * np.array([[1, 2], [3, 4]])
```

```
---------------------------------------------------------------------------
```

```
ValueError                                Traceback (most recent call last)
/Users/mrdavis/projects/numpy_tutorial_2012-05-31/<ipython-input-53-b5bd2537fe31> in <module>()
----> 1 a * np.array([[1, 2], [3, 4]])

ValueError: shape mismatch: objects cannot be broadcast to a single shape
```

4. NumPy Functions

http://docs.scipy.org/doc/numpy/reference/ufuncs.html#available-ufuncs

In [54]:
```
a
```

Out[54]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

In [56]:
```
np.sin(a)
```

Out[56]:
```
array([[ 0.        ,  0.84147098],
       [ 0.90929743,  0.14112001],
       [-0.7568025 , -0.95892427],
       [-0.2794155 ,  0.6569866 ]])
```

5. Array Attributes & Methods

http://docs.scipy.org/doc/numpy/reference/arrays.ndarray.html#array-attributes

http://docs.scipy.org/doc/numpy/reference/arrays.ndarray.html#array-methods

In [57]:
```
a
```

Out[57]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

In [58]:
```
a.shape
```

Out[58]:
```
(4, 2)
```

In [59]:
```
a.size
```

Out[59]:
```
8
```

In [60]:
```
a.dtype
```

Out[60]:
```
dtype('int64')
```

In [61]:
```
a.nbytes
```

Out[61]:
```
64
```

```
In [62]: a.reshape((2,4))

Out[62]: array([[0, 1, 2, 3],
                [4, 5, 6, 7]])

In [63]: a.argsort()

Out[63]: array([[0, 1],
                [0, 1],
                [0, 1],
                [0, 1]])

In [67]: a.min()

Out[67]: 0

In [68]: a.max()

Out[68]: 9

In [69]: print a.mean(), a.sum(), a.std(), a.prod()

         4.5 45 2.87228132327 0

In [71]: np.mean([1, 2, 3, 4])

Out[71]: 2.5
```

6. numpy.random

http://docs.scipy.org/doc/numpy/reference/routines.random.html

```
In [110]: np.random.permutation(np.arange(10))

Out[110]: array([2, 3, 5, 6, 8, 1, 0, 7, 9, 4])

In [77]: np.random.random((2, 2))

Out[77]: array([[ 0.50411342,  0.10708051],
                [ 0.7061658 ,  0.87775115]])

In [79]: np.random.random_integers(5, 10, 5)

Out[79]: array([7, 7, 5, 8, 9])
```

7. Masked Arrays

http://docs.scipy.org/doc/numpy/reference/maskedarray.html

http://docs.scipy.org/doc/numpy/reference/routines.ma.html

```
In [80]: import numpy.ma as ma

In [81]: a = ma.masked_greater(np.random.random(10), 0.5)
In [82]: a
```

```
Out[82]: masked_array(data = [0.126459538933 -- -- 0.382754789138 0.0975191738884
0.182983512924
 0.367432667685 -- 0.0585601591978 --],
             mask = [False  True  True False False False False  True False  True],
       fill_value = 1e+20)
```

```
In [84]: print a.mean(), a.max()
```

```
0.202618306961 0.382754789138
```

```
In [85]: a[0] = ma.masked
         print a
```

```
[-- -- -- 0.382754789138 0.0975191738884 0.182983512924 0.367432667685 --
 0.0585601591978 --]
```

```
In [87]: a[0] = 1.0
         print a
```

```
[1.0 -- -- 0.382754789138 0.0975191738884 0.182983512924 0.367432667685 --
 0.0585601591978 --]
```

```
In [88]: a.filled()
```

```
Out[88]: array([  1.00000000e+00,   1.00000000e+20,   1.00000000e+20,
                  3.82754789e-01,   9.75191739e-02,   1.82983513e-01,
                  3.67432668e-01,   1.00000000e+20,   5.85601592e-02,
                  1.00000000e+20])
```

8. Array Comparison

http://docs.scipy.org/doc/numpy/reference/routines.testing.html

```
In [89]: a = np.arange(10)
         b = np.arange(5, 25, 2)
```

```
In [90]: b.shape
```

```
Out[90]: (10,)
```

```
In [91]: print a
         print b
```

```
[0 1 2 3 4 5 6 7 8 9]
[ 5  7  9 11 13 15 17 19 21 23]
```

```
In [93]: a == b
```

```
Out[93]: array([False, False, False, False, False, False, False, False, False, False],
dtype=bool)
```

```
In [94]: np.allclose(a, b)
```

```
Out[94]: False
```

```
In [95]: a[1] = 7
```

```
In [97]: (a == b).any()
```

```
Out[97]: True
```

```
In [98]: (a == b).all()
```

```
Out[98]: False
```

```
In [99]: np.testing.assert_allclose(a, b)
```

```
---------------------------------------------------------------------------
AssertionError                            Traceback (most recent call last)
/Users/mrdavis/projects/numpy_tutorial_2012-05-31/<ipython-input-99-b55a17f72509> in
<module>()
----> 1 np.testing.assert_allclose(a, b)

/usr/stsci/pyssgdev/2.7/numpy/testing/utils.pyc in assert_allclose(actual, desired,
rtol, atol, err_msg, verbose)
   1128     header = 'Not equal to tolerance rtol=%g, atol=%g' % (rtol, atol)
   1129     assert_array_compare(compare, actual, desired, err_msg=str(err_msg),
-> 1130                          verbose=verbose, header=header)
   1131
   1132 def assert_array_almost_equal_nulp(x, y, nulp=1):

/usr/stsci/pyssgdev/2.7/numpy/testing/utils.pyc in assert_array_compare(comparison,
x, y, err_msg, verbose, header)
    616                                 names=('x', 'y'))
    617             if not cond :
--> 618                 raise AssertionError(msg)
    619     except ValueError:
    620         msg = build_err_msg([x, y], err_msg, verbose=verbose, header=header,

AssertionError:
Not equal to tolerance rtol=1e-07, atol=0

(mismatch 100.0%)
 x: array([0, 7, 2, 3, 4, 5, 6, 7, 8, 9])
 y: array([ 5,  7,  9, 11, 13, 15, 17, 19, 21, 23])
```

```
In [ ]:
```