Numpy

Python for Ecologists

Tao Hong, Tom Purucker, Jonathan Flaishans, Marcia Snyder

> Ecological Society of America Workshop Minneapolis, MN hongtao510@gmail.com

> > August 1, 2013

_						
	١,	10	r١	71	\triangle	١٨/

- Install Numpy
- Array
- Index
- Random number
- Basic operations
- Reference

Install Numpy

- Windows
 - Official

http://sourceforge.net/projects/numpy/?source=dlp

Unofficial (Windows binaries)

http://www.lfd.uci.edu/~gohlke/pythonlibs/

■ Portable Python

http://portablepython.com/wiki/PortablePython2.7.5.1/

Arrays

■ Create an array

```
a = np.array([10, 20, 30, 40], float)
[ 10. 20. 30. 40.]

b = np.array([10, 20, 30, 40])
[10 20 30 40]

c = np.array([[1, 2],[4, 5]])
[[1 2]
[4 5]]

d = np.array([[1, 2],[4, 5.0]])
[[ 1. 2.]
[ 4. 5.]]
```

Notes			

Notes			
-			

Notes			

Notes			

np.arange np.array is not efficient, try np.arange np.arange(start, stop, step, dtype=None) f = np.arange(0,4,1) #does not include '4' [0 1 2 3]

```
Notes
```

np.linespace

■ Compare to np.arange

g = np.arange(4)[0 1 2 3]

np.linspace(start, stop, num, endpoint=True, retstep=False)

Notes

Other ways to create arrays

■ np.ones

```
x = np.ones((2,3), float)
[[ 1. 1. 1.]
[ 1. 1. 1.]]
```

■ np.zeros

```
x = np.zeros((2,3), float)
[[ 0. 0. 0.]
[ 0. 0. 0.]]
```

■ np.identity

```
x = np.identity(3, float)
[[ 1.  0.  0.]
[ 0.  1.  0.]
[ 0.  0.  1.]]
```

Notes ______

Array properties (1)

```
a = np.array([[1, 2, 3], [4, 5, 6]], float)
a.shape #shape
(2, 3)

a.reshape(1,6)
[[ 1. 2. 3. 4. 5. 6.]]

a.ndim #number of dimensions
2

a.dtype #data type
float64

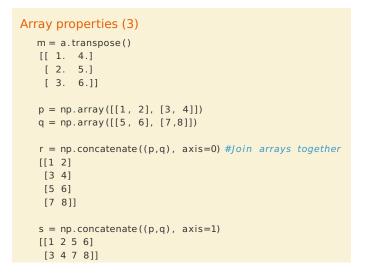
a.size #number of elements
6
```

Notes			

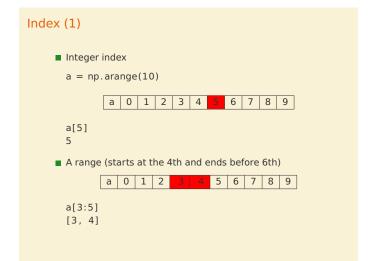
a = np.array([[1, 2, 3], [4, 5, 6]], float) k = a.flatten() [1. 2. 3. 4. 5. 6.] I = a.tolist() #array to list [[1.0, 2.0, 3.0], [4.0, 5.0, 6.0]] type(I) <type 'list'>

Array properties (2)

```
Notes ______
```



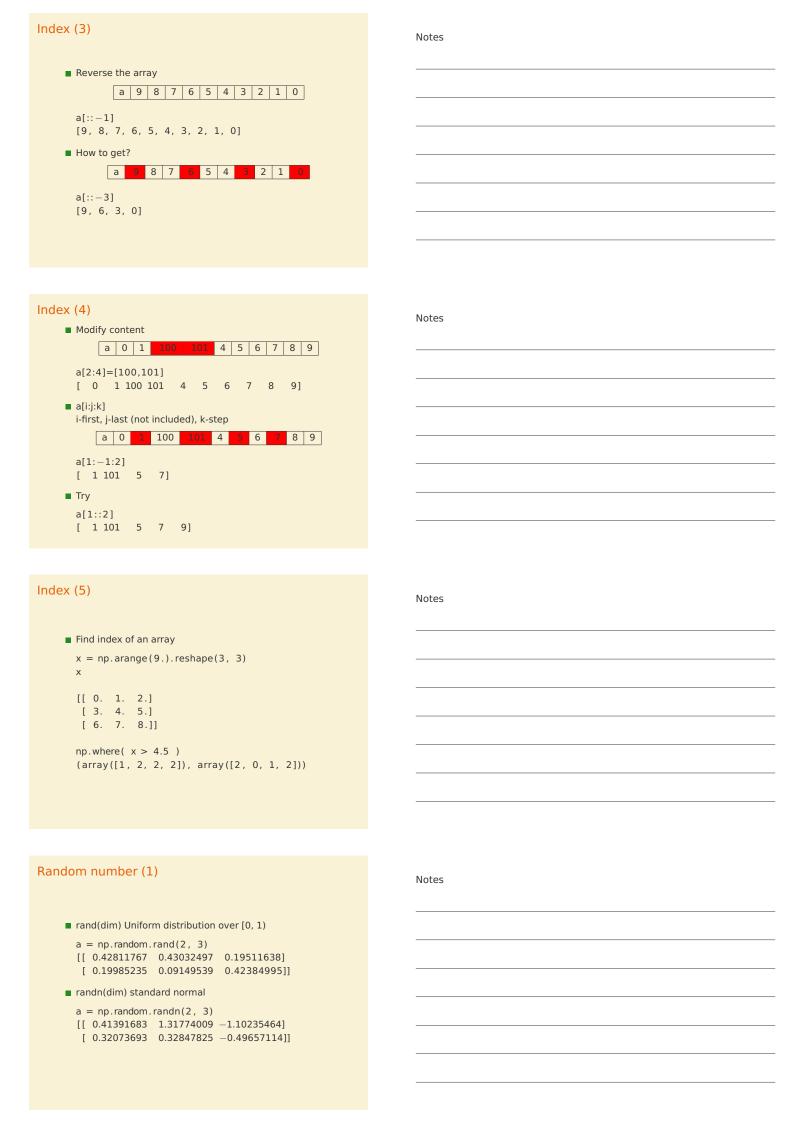






Index (2) The first three elements a 0 1 2 3 4 5 6 7 8 9 a[:3] [0,1,2] Counting backwards a 0 1 2 3 4 5 6 7 8 9 a[:-7] [0,1,2]

Notes			



Random number (2) log-normal lognormal(mean, sigma, dim) Possion poisson(mean, dim) Beta beta(a, b, dim) Fix a seed seed(number) more distritions are available http://docs.scipy.org/doc/numpy/reference/routines.random.html

```
Basic operations (1)

    sum
    a = np.array([[1, 2, 3], [4, 5, 6]], float)
    [[ 1. 2. 3.]
    [ 4. 5. 6.]]
    a.sum()
    21.0
    a.sum(axis=0) #col sum
    [ 5. 7. 9.]

    mean
    a.mean()
    3.5

    variance
    a.var()
    2.916666666667
```

```
Basic operations (3)

■ inverse

a = np.array([[1, 2], [4, 5]], float)
[[ 1. 2.]
[ 4. 5.]]

b=np.linalg.inv(a)
[[-1.666666667 0.66666667]
[ 1.33333333 -0.33333333]]

■ determinant
    np.linalg.det(a)
    -3
```

Notes			
Notes			
Notes			
Notes			

Basic operations (4) matrix multiply np.dot(a,b) [[1. 0.] [0. 1.]] ■ element-wise multiply a∗b [[-1.66666667 1.33333333] [5.33333333 -1.66666667]] solve a linear system a x c=b c=np.linalg.solve(a,b) [[3.66666667 -1.33333333] [-2.66666667 1.]]

```
Notes
```

Shallow copy

■ arrays share the same elements

```
a = np.arange(0, 60, 10)
[ 0 10 20 30 40 50]
[ 0 10 20 30 40 50]
a[0]=100
[100 10 20 30 40 50]
[100 10 20 30 40 50]
```

Notes

Deep copy

■ each array has its own elements

```
a = np.arange(0, 60, 10)
import copy
b= copy.deepcopy(a)
[ 0 10 20 30 40 50]
[ 0 10 20 30 40 50]
a[0]=100
[100 10 20 30 40 50]
[0 10 20 30 40 50]
```

Notes			

Reference

- Official document http://docs.scipy.org/doc/
- NumPy for MATLAB users http://www.scipy.org/NumPy_for_Matlab_Users
- NumPy for R (and S-Plus) users http://mathesaurus.sourceforge.net/r-numpy.html

1 11		5	. ,
Stackoverflow			
httn://stackoverflow	com/		

Notes			