Practical Computing for Scientists

Armin Sobhani
CSCI 2000U
UOIT – Fall 2015

UNIVERSITY OF ONTARIO INSTITUTE OF TECHNOLOGY

Python

NumPy (Continue)

UNIVERSITY OF ONTARIO INSTITUTE OF TECHNOLOGY

Other Ways to Create Arrays

```
>>> np.arange(5, dtype=float)
array([ 0., 1., 2., 3., 4.])
```

Other Ways to Create Arrays

```
>>> np.arange(5, dtype=float)
array([ 0., 1., 2., 3., 4.])
>>> np.arange(1, 6, 2, dtype=int)
array([1, 3, 5])
```

Other Ways to Create Arrays

```
>>> np.ones((2,3), dtype=float)
array([[ 1., 1., 1.],
       [ 1., 1., 1.]])
```

Other Ways to Create Arrays

```
>>> np.ones((2,3), dtype=float)
array([[ 1., 1., 1.],
       [ 1., 1., 1.]])
>>> np.zeros(7, dtype=int)
array([0, 0, 0, 0, 0, 0, 0])
```

Other Ways to Create Arrays

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

Other Ways to Create Arrays

```
>>> a = np.array([[1, 2, 3], [4, 5, 6]], float)
>>> np.zeros_like(a)
array([[ 0., 0., 0.],
       [ 0., 0., 0.]])
```

Other Ways to Create Arrays

```
>>> a = np.array([[1, 2, 3], [4, 5, 6]], float)
>>> np.zeros_like(a)
array([[ 0., 0., 0.],
       [ 0., 0., 0.]])
>>> np.ones_like(a)
array([[ 1., 1., 1.],
       [ 1., 1., 1.]])
```

Other Ways to Create Arrays

```
>>> np.identity(4, dtype=float)
array([[ 1., 0., 0., 0.],
       [ 0., 1., 0., 0.],
       [ 0., 0., 1., 0.],
       [ 0., 0., 0., 1.]])
```

Other Ways to Create Arrays

```
>>> np.identity(4, dtype=float)
array([[ 1., 0., 0., 0.],
       [ 0., 1., 0., 0.],
       [ 0., 0., 1., 0.],
       [ 0., 0., 0., 1.]])
>>> np.eye(4, k=1, dtype=float)
array([[ 0., 1., 0., 0.],
       [ 0., 0., 1., 0.],
       [ 0., 0., 0., 1.],
       [ 0., 0., 0., 0.]])
```

Array Mathematics

```
>>> a = np.array([1,2,3], float)
>>> b = np.array([5,2,6], float)
```

Array Mathematics

```
>>> a = np.array([1,2,3], float)
>>> b = np.array([5,2,6], float)
>>> a + b
array([6., 4., 9.])
```

Array Mathematics

```
>>> a = np.array([1,2,3], float)
>>> b = np.array([5,2,6], float)
>>> a + b
array([6., 4., 9.])
>>> a - b
array([-4., 0., -3.])
```

Array Mathematics

```
>>> a = np.array([1,2,3], float)
>>> b = np.array([5,2,6], float)
>>> a + b
array([6., 4., 9.])
>>> a - b
array([-4., 0., -3.])
>>> a * b
array([5., 4., 18.])
```

Array Mathematics

```
>>> a = np.array([1,2,3], float)
>>> b = np.array([5,2,6], float)
>>> a + b
array([6., 4., 9.])
>>> a - b
array([-4., 0., -3.])
>>> a * b
array([5., 4., 18.])
>>> b / a
array([5., 1., 2.])
```

Array Mathematics

```
>>> a = np.array([1,2,3], float)
>>> b = np.array([5,2,6], float)
>>> a + b
array([6., 4., 9.])
>>> a - b
array([-4., 0., -3.])
>>> a * b
array([5., 4., 18.])
>>> b / a
array([5., 1., 2.])
>>> a % b
array([1., 0., 3.])
```

Array Mathematics

```
>>> a = np.array([1,2,3], float)
>>> b = np.array([5,2,6], float)
>>> a + b
array([6., 4., 9.])
>>> a - b
array([-4., 0., -3.])
>>> a * b
array([5., 4., 18.])
>>> b / a
array([5., 1., 2.])
>>> a % b
array([1., 0., 3.])
>>> b**a
array([5., 4., 216.])
```

Array Mathematics

```
>>> a = np.array([1,2,3], float)
>>> b = np.array([4,5], float)
```

Array Mathematics

```
>>> a = np.array([1,2,3], float)
>>> b = np.array([4,5], float)
>>> a + b
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: shape mismatch: objects cannot be broadcast to a
single shape
```

Array Mathematics

```
>>> a = np.array([[1,2], [3,4]], float)
>>> b = np.array([[2,0], [1,3]], float)
>>> a * b
```

Array Mathematics

```
>>> a = np.array([[1,2], [3,4]], float)
>>> b = np.array([[2,0], [1,3]], float)
>>> a * b
array([[2., 0.],
       [3., 12.]])
```

Array Mathematics

```
>>> a = np.array([[1, 2], [3, 4], [5, 6]], float)
>>> b = np.array([-1, 3], float)
```

Array Mathematics

```
>>> a = np.array([[1, 2], [3, 4], [5, 6]], float)
>>> b = np.array([-1, 3], float)
>>> a
array([[ 1., 2.],
       [ 3., 4.],
       [ 5., 6.]])
>>> b
array([-1., 3.])
```

Array Mathematics

```
>>> a = np.array([[1, 2], [3, 4], [5, 6]], float)
>>> b = np.array([-1, 3], float)
>>> a
array([[ 1., 2.],
       [ 3., 4.],
       [ 5., 6.]])
>>> b
array([-1., 3.])
>>> a + b
```

Array Mathematics

```
>>> a = np.array([[1, 2], [3, 4], [5, 6]], float)
>>> b = np.array([-1, 3], float)
>>> a
array([[ 1., 2.],
       [ 3., 4.],
       [ 5., 6.]])
>>> b
array([-1., 3.])
>>> a + b
```

broadcasted:

```
array([[-1., 3.],
       [-1., 3.],
       [-1., 3.]])
```

Array Mathematics

```
>>> a = np.array([[1, 2], [3, 4], [5, 6]], float)
>>> b = np.array([-1, 3], float)
>>> a
array([[ 1., 2.],
       [ 3., 4.],
       [ 5., 6.]])
>>> b
array([-1., 3.])
>>> a + b
array([[ 0., 5.],
       [ 2., 7.],
       [ 4., 9.]])
```

broadcasted:

```
array([[-1., 3.],
       [-1., 3.],
       [-1., 3.]])
```

Array Mathematics

```
>>> a = np.array([1, 4, 9], float)
```

Array Mathematics

```
>>> a = np.array([1, 4, 9], float)
>>> np.sqrt(a)
array([ 1., 2., 3.])
```

Array Mathematics

```
>>> a = np.array([1, 4, 9], float)
>>> np.sqrt(a)
array([ 1., 2., 3.])
```

abs	arctanh	sign
arccos	cos	sin
arcsin	cosh	sinh
arctan	exp	sqrt
arccosh	log	tan
arcsinh	log10	tanh

Array Mathematics

```
>>> a = np.array([1.1, 1.5, 1.9], float)
>>> np.floor(a) # lower integer
array([ 1., 1., 1.])
```

Array Mathematics

```
>>> a = np.array([1.1, 1.5, 1.9], float)
>>> np.floor(a) # lower integer
array([ 1., 1., 1.])
>>> np.ceil(a) # upper integer
array([ 2., 2., 2.])
```

Array Mathematics

```
>>> a = np.array([1.1, 1.5, 1.9], float)
>>> np.floor(a) # lower integer
array([ 1., 1., 1.])
>>> np.ceil(a) # upper integer
array([ 2., 2., 2.])
>>> np.rint(a) # nearest (rounded) integer
array([ 1., 2., 2.])
```

Array Mathematics

```
>>> np.pi
3.1415926535897931
>>> np.e
2.7182818284590451
```

Array Iteration

```
>>> a = np.array([1, 4, 5], int)
>>> for x in a:
        print(x)
```

Array Iteration

```
>>> a = np.array([1, 4, 5], int)
>>> for x in a:
        print(x)
1
4
5
```

Array Iteration

```
>>> a = np.array([[1, 2], [3, 4], [5, 6]], float)
>>> for x in a:
        print(x)
```

Array Iteration

```
>>> a = np.array([[1, 2], [3, 4], [5, 6]], float)
>>> for x in a:
        print(x)
[ 1. 2.]
[ 3. 4.]
[ 5. 6.]
```

Array Iteration

```
>>> a = np.array([[1, 2], [3, 4], [5, 6]], float)
>>> for (x, y) in a:
        print(x * y)
```

Array Iteration

```
>>> a = np.array([[1, 2], [3, 4], [5, 6]], float)
>>> for (x, y) in a:
        print(x * y)
2.0
12.0
30.0
```