Python NumPy

[Python][Python matplotlib]

・http://www.numpy.org/

ドキュメント

・http://docs.scipy.org/doc/

■リファレンス

・https://docs.scipy.org/doc/numpy-1.14.2/genindex.html

概要

■ NumPy

・http://ja.wikipedia.org/wiki/NumPy
・大部分が C で書かれた Python 用の拡張モジュール
・数値配列、行列を定義し、基本的な操作が可能

■ SciPy

・http://ja.wikipedia.org/wiki/SciPy
・先進的な数学を NumPy を使って行うための Python のライブラリ
・信号処理、最適化、統計、など

■ matplotlib

・出版物のクオリティで、インタラクティブにプロットするのを容易にするライブラリ

NumPy

・http://www.scipy.org/Tentative_NumPy_Tutorial

・NumPy は多次元配列を扱うライブラリで、主に扱うデータ型は配列である
・配列は同じ型の要素のセットであり、正の整数のベクターによりインデックス付けされる

■ Install

・http://www.scipy.org/Installing_SciPy/Linux

apt-get を使って ubuntu へインストール

```
sudo apt-get install python-numpy python-scipy
```

PIP を使ってインストール

```
# pip install numpy
```

PIP を使って Windows にインストール

・Microsoft Visual C++ Compiler for Python 2.7
・http://www.microsoft.com/en-us/download/confirmation.aspx?id=44266

```
C:\Python27\Scripts>pip install numpy
```

配列の生成

■リストから生成

```
>>> from numpy import *
>>> a = array( [ 10, 20, 30, 40 ] )
>>> a
array([10, 20, 30, 40])
```

■ arange を使い配列を生成

```
arange([start,] stop[, step,], dtype=None)
```

0 から始まる整数の配列を生成

```
>>> b = arange(4)
>>> b
array([0, 1, 2, 3])
```

0 から 3 まで 0.5 きざみの配列を生成

```
>>> np.arange(0,3,0.5)
array([ 0. ,  0.5,  1. ,  1.5,  2. ,  2.5])
```

■等しく割り付けられた配列を作成

```
>>> c = linspace(-pi,pi,3)
>>> c
array([-3.14159265,  0.        ,  3.14159265])
```

■既存の配列から生成

```
>>> a1 = array([10,20,30])
>>> a2 = array([1,2,3])
>>> a3 = (a1 + a2) * 2
>>> a3
array([22, 44, 66])
```

■多次元配列

```
>>> x = ones( (3,4) )
>>> x
array([[ 1.,  1.,  1.,  1.],
       [ 1.,  1.,  1.,  1.],
       [ 1.,  1.,  1.,  1.]])
>>> x.shape # タプルで次元を取得
(3, 4)
```

■既存の配列の次元を変更

```
>>> y = arange(12)
>>> y
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11])
>>> y.shape = (3,4)
>>> y
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

参照

・https://qiita.com/supersaiakujin/items/d63c73bb7b5aac43898a

■箇所を指定

1 次元

```
x[n]
```

2 次元

```
x[n,m]
```

■範囲を指定

1 次元

```
x[start:end:step]
```

2 次元

```
x[start:end:step,start:end:step]
```

■行を抽出

```
x[r]
x[r,]
x[r,:]
```

■列を抽出

```
x[,:c]
```

取り出した値が 1 次元の配列になるため注意 reshape()

■条件を満たすデータを取り出す

操作

■次元が異なる配列の演算

それぞれの列に掛ける

```
>>> x = arange(4)
>>> x
array([0, 1, 2, 3])
>>> x * 2
array([0, 2, 4, 6])
```

それぞれの行に足し込む

```
>>> y = arange(10)
>>> y.shape = (2,5)
>>> y
array([[0, 1, 2, 3, 4],
       [5, 6, 7, 8, 9]])
>>> y1 = array([10,20,30,40,50])
>>> y2 = y + y1
>>> y2
array([[10, 21, 32, 43, 54],
       [15, 26, 37, 48, 59]])
```