

Numpy 陣列的定義與屬性



簡報閱讀



範例與作業



問題討論



學習心得(完成)



陪跑專家：James/Hong/維元

重要知識點



- NumPy介紹與安裝
- 認識 Numpy 套件與 Array 陣列定義
- 正確使用 Array 的常用屬性
- 能夠掌握 Array 的重要特性

with Python

「NumPy 是 Python 語言的一個擴充程式庫。支援高階大量的維度陣列與矩陣運算，此外也針對陣列運算提供大量的數學函式庫。」 - 維基百科

NumPy



The fundamental package for scientific computing with Python

GET STARTED

NumPy v1.20.0 Type annotation support - Performance improvements through multi-platform SIMD

POWERFUL N-DIMENSIONAL ARRAYS

Fast and versatile, the NumPy vectorization, indexing, and broadcasting concepts are the de-facto standards of array computing today.

NUMERICAL COMPUTING TOOLS

NumPy offers comprehensive mathematical functions, random number generators, linear algebra routines, Fourier transforms, and more.

INTEROPERABLE

NumPy supports a wide range of hardware and computing platforms, and plays well with distributed, GPU, and sparse array libraries.

PERFORMANT

The core of NumPy is well-optimized C code. Enjoy the flexibility of Python with the speed of compiled code.

EASY TO USE

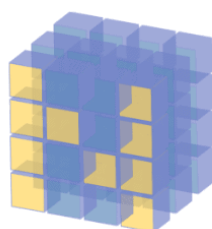
NumPy's high level syntax makes it accessible and productive for programmers from any background or experience level.

OPEN SOURCE

Distributed under a liberal BSD license, NumPy is developed and maintained publicly on GitHub by a vibrant, responsive, and diverse community.

NumPy 簡介

- NumPy 是廣受歡迎的 open source Python 程式庫，支援多維度陣列與矩陣運算，也針對陣列運算提供大量的數學函式庫。
- NumPy 的核心是由 C 語言開發，所以在陣列與矩陣運算時具有高效能的特性。
- 2019 年時 GitHub 統計指出有 74% 的機器學習專案中使用 NumPy。



NumPy



NumPy 是一個提供 Python 用於科學計算開源項目，最早是 2005 年基於 Numeric 和 Numarray 發展而來。效率是 NumPy 重要的特性，主要用於陣列/矩陣的計算。綜合來說，Numpy 具備以下特性：

1. 強大的多維類別類型
2. 矩陣廣播運算的特性
3. 底層集成 C/C++ 和 Fortran 實作
4. 大量科學/數學計算的實作函數（線性代數、三角函數 ...）

在程式中載入 Numpy

不管是在 .py 或是 Jupyter 的環境，都可以利用 import 的方法載入 numpy 函式庫，一般習慣命名成 np：

```
1 import numpy as np
```

載入成功後，可以正確印出套件的物件與版本：

```
1 print(np) # <module 'numpy' from '../numpy/_  
2 print(np.__version__) # 1.20.1
```

建立 NumPy array (陣列)

- 要用序列數字產生等差一維陣列的話，可以使用 `arange()` 與 `linspace()` 函式，兩者的函式引數很類似，其中結束值為必填，起始

的表示方式。

`arange()` 不同於 `linspace()` 在於對產生的元素可以有[更多的控制](#)。

參考資料：[range\(\)](#)、[np.arange\(\)](#)、[np.linspace\(\)](#) 的差異

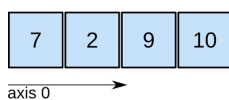
- 建立陣列的方式是透過執行 NumPy 函式，依照不同的目的，以下逐一介紹常用來建立陣列的函式。

最基本的方式是呼叫 `array()` 函式，可將 Python list 或元組 (`tuple`) 的值建立為 NumPy array。

NdArray

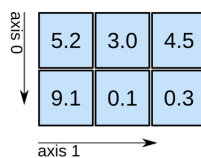
NumPy 提供了一個[同類型元素](#)的[多維容器](#)型態，是一個所有元素（通常是數字）的類型都相同的矩陣類容器，並通過正整數索引。

1D array



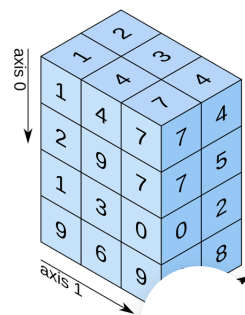
shape: (4,)

2D array



shape: (2, 3)

3D array



shape: (4, 3, 3)

從一個陣列開始

type() 查看其類型。

```
1 import numpy as np
2 a = np.arange(15).reshape(3, 5)
3 print(a)
4 # array([[ 0,  1,  2,  3,  4],
5 #        [ 5,  6,  7,  8,  9],
6 #        [10, 11, 12, 13, 14]])
7
8 print(type(a)) # <type 'numpy.ndarray'>
```

補充：這個範例是先產生一個陣列做查看，具體建立陣列的方式後面會提到。

陣列的常用屬性

NumPy 的數組被稱為 ndarray 或簡稱 Array，提供以下屬性：

- ndarray.ndim
- ndarray.shape
- ndarray.size
- ndarray.dtype
- ndarray.itemsize
- ndarray.data

```
1 print(a.ndim)
2 print(a.shape)
3 print(a.size)
4 print(a.dtype)
5 print(a.itemsize)
6 print(a.data)
```

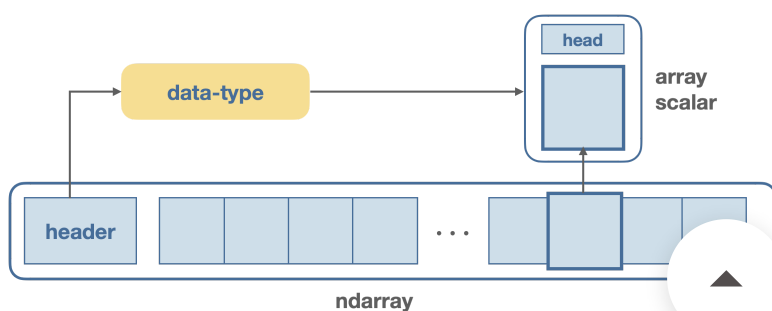
接下來，來看一下這些屬性的意義：

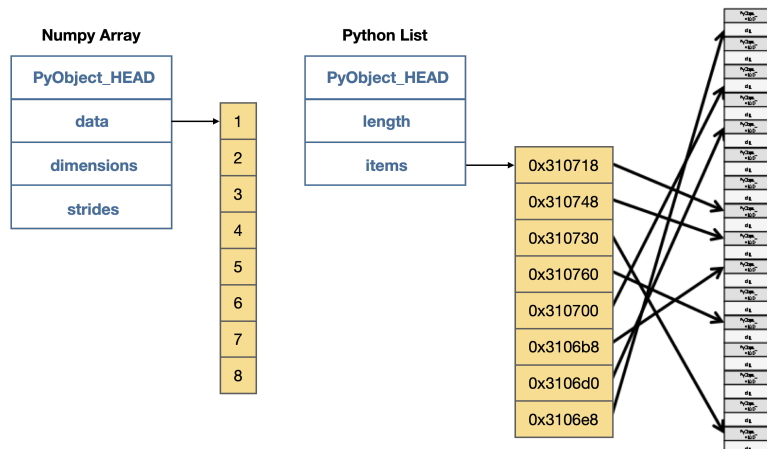
- `ndarray.ndim`：陣列有多少維度
- `ndarray.shape`：每個維度的大小
- `ndarray.size`：陣列當中有幾個元素
- `ndarray.dtype`：陣列中的資料型態
- `ndarray.itemsize`：陣列中每個元素佔用的空間
- `ndarray.data`：陣列所存在的記憶體位置

補充：除此之外，其他完整屬性可以從 [官網文件](#) 中找到。

Python 內建容器 與 NumPy 陣列的比較

Python 內建的列表（List）是由獨立的元素所組成的容器，元素的型態可以不同。而 NumPy 陣列是同質性的（homogeneous）的物件，也就是每個元素的型態都必須相同。因此在底層定義與操作略有差異。





List 與 Array 的轉換

NumPy 的陣列與 Python 的 List 轉換可以有列兩種方法，請思考一下這兩者有什麼差異：

`list(a)` 只會把第一層的元素轉換成 List，多層的話只有第一層會轉；

`tolist()` 才能達成多層的型態轉換。

```
1 list(a)
2 # [array([0, 1, 2, 3, 4]), array([5, 6, 7, 8,
3
4 a.tolist()
5 # [[0, 1, 2, 3, 4], [5, 6, 7, 8, 9], [10, 11,
```

陣列型態所代表的數學意義

陣列與列表的差異在程式語言的資料結構中，久，而最初 Python 會選擇以列表作為基本器，是考量到 Python 對於型態的彈性。在數學中最基本的運算單位是向量（有方向的數字集合），

補 Python 列表的不足。

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \dots \\ a_{2,1} & a_{2,2} & a_{2,3} & \dots \\ a_{3,1} & a_{3,2} & a_{3,3} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ \vdots \end{bmatrix}$$

應該使用 Numpy 的原因

最後，我們整理出四個 Numpy 更強大的特性：

- Numpy 使用更少的空間儲存資料
- Numpy 對於元素的查找更加快速高效
- Numpy 可以更好地表示數學中的向量/矩陣
- Numpy 支援了矩陣相關的數學運算

更具體來說，NumPy 陣列不只是程式設計當中的陣列，更滿足了數學當中的矩陣特性。

知識點回顧

- NumPy 介紹與安裝
- 認識 Numpy 套件與 Array 陣列定義
- 正確使用 Array 的常用屬性
- 能夠掌握 Array 的重要特性

參考資料

[Numpy官方網站](#)



檔。對於任何函式不熟的時候，都可以從文件中找到更深入的用法。

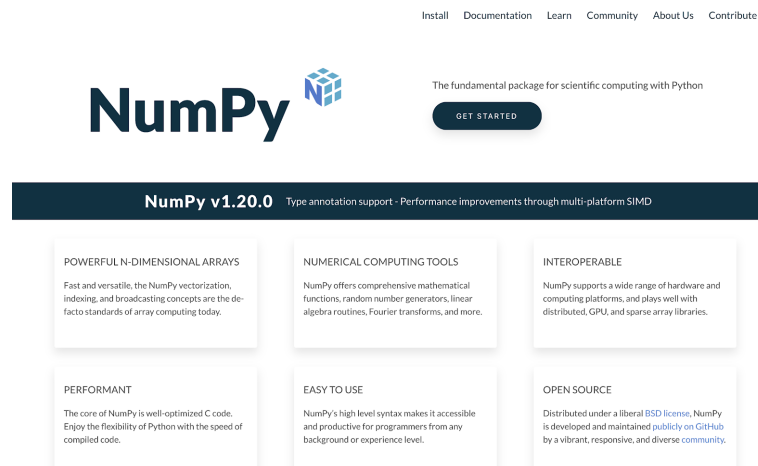
重要知識點

NumPy: The fundamental packag...

NumPy 簡介

NumPy 的重要特性

在程式中載入 Numpy



從零開始學資料科學：Numpy 基礎入門

網站：[techbridge](https://techbridge.tw)

本系列文章將透過系統介紹資料科學（Data Science）相關的知識，透過 Python 帶領讀者從零開始進入資料科學的世界。這邊我們將介紹 Numpy 這個強大的 Python 函式庫。



[下一步：閱讀範例與完成作業](#)



[AI共學社群](#)

[我的](#)



9+

