程式設計概論
PROGRAMMING 101
—NUMPY進行資料運算

授課老師：邱淑怡

DATE:5/4/2021

2 OUTLINE

- NumPy (Numeric Python)
  - NumPy (https://numpy.org/)為第三方套件，需安裝
- NumPy模組的操作

3 NUMPY簡介

- NumPy (Numerical Python)是運用Python進行科學計算的基本套件(模組)
- 可以定義任何資料型態
  - 強大的N維陣列物件
  - 提供資料的高效能多維容器(multi-dimensional container)
  - 能快速地與各種資料庫整合
  - 統計功能
  - 複雜的廣播(broadcasting)功能
  - 用於整合C/C++與Fortran程式的工具
  - 實用的線性代數、傅立葉轉換及亂數功能(依據資料分布產生數值)

4 N維陣列

- NumPy最顯著的特徵在於它提供的資料結構是陣列(array)
- 陣列可以定義維度，因此陣列的全稱是N維陣列(ndarray), N是一維、二維或多維
- 陣列的結構
  - 一維陣列 np.array(一個數列)
  - 二維陣列 np.array([數列1,數列2,…,數列n])
  - 其中小括號的數列可以是一個數列(相當於1*n的向量)
  - m個數列作為元素所組成的一個數列(相當於m*n的向量)

5 WHAT THE DIFFERENCE BETWEEN A NUMPY ARRAY AND A PYTHON LIST?

- Python list can contain different data types within a single list, all of the elements in a NumPy array should be homogeneous.
- NumPy arrays are faster and more compact than Python lists.
- An array consumes less memory and is convenient to use. NumPy uses much less memory to store data and it provides a mechanism of specifying the data types.

6 一維陣列

◆使用array()函式從list or tuple建立一維陣列

```
import numpy as np
A=np.array([10,20,40,76,50,26,3])
print(A)
print(A[0])
print(A[2:5])
print(A[:4])
print(A[2:])
A[[0,1]]=70  #modify A[0,1]
```

元素	值
A[0]	10
A[1]	20
A[2]	40

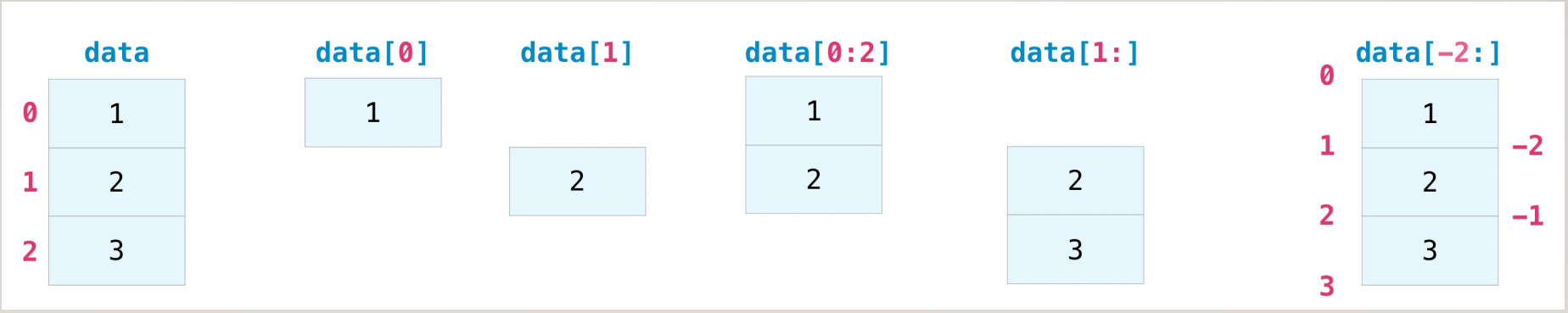
為何要用ndarray，何不直接用list或tuple?
理由: ndarray處理大量資料較快，因為它會將資料存在記憶體連續區塊，但list將資料存在非連續區塊。
(1) NumPy建立陣列時已具固定大小；list大小是動態的
(2) NumPy陣列的元素必須是相同型別；list元素無此限制
(3) NumPy陣列針對大量資料提供進階且高速的數學運算

7 ARRAYS

data = np.array([1, 2, 3)]

data[1]

data[-2:]



8 一維陣列(CONT.)

- import numpy as np
- print(np.zeros(5)) #array([0. 0. 0. 0. 0.])
- print(np.ones(4)) # array([1. 1. 1. 1.])
- print(np.empty(3))
- print(np.arange(start=3, stop=10, step=3, dtype=int)) # get array([3 6 9])
- print(np.arange(4,dtype=int))   # get array([0 1 2 3])
- print(np.arange(0,2,0.4)) #get array([0. 0.4 0.8 1.2 1.6])
- print(np.linspace(0,10,6))

linspace產生等差序列的函式，第一參數是起始值，第二參數是終止值(預設為包含在內)，第三參數是陣列中元素的個數

9 一維陣列操作

```
A=np.array([10,30,25,42,55,96,45])
B=np.array([78,82,93])
C=np.insert(A,2,100)    # 陣列A中索引2處插入100並將結果指派給C
D=np.insert(A,[1,4],-10) # 陣列A中索引1,4 處插入-10並將結果指派給D
E=np.delete(A,4)
F=np.delete(A,[2,3])
G=np.concatenate((A,B))  # 結合A 與B 陣列後並指派給G
H=np.concatenate((A,[20,22]))
```

10 一維陣列的練習題

1. 透過NumPy產生包含0-9整數的陣列
2. 透過NumPy產生從1-14且每一步進值為3的陣列
3. 透過NumPy產生從0-100並且元素個數為51的等差陣列

將上述陣列印出

11 一維陣列的練習題_參考程式

1. 透過NumPy產生包含0-9整數的陣列
2. 透過NumPy產生從1-14且每一步進值為3的陣列
3. 透過NumPy產生從0-100並且元素個數為51的等差陣列

將上述陣列印出

```
import numpy as np
A1=np.arange(10)
print(A1)
A2=np.arange(1,15,3)
print(A2)
A3=np.linspace(0,100,51)
print(A3)
```

12 二維陣列

- 一維陣列是呈線性的一度空間；二維陣列是呈平面的二度空間

```
Grades=np.array([[96,65,73],[88,76,82],[92,84,89],[82,73,64],[70,83,68]])
Grades是5*3的二維陣列
```

	國文	英文	數學
學生1	96	65	73
學生2	88	76	82
學生3	92	84	89
學生4	82	73	64
學生5	70	83	68

對應的索引值

	國文	英文	數學
學生1	[0, 0]	[0, 1]	[0, 2]
學生2	[1, 0]	[1, 1]	[1, 2]
學生3	[2, 0]	[2, 1]	[2, 2]
學生4	[3, 0]	[3, 1]	[3, 2]
學生5	[4, 0]	[4, 1]	[4, 2]

13 檢視陣列的屬性

函數名	功能	Python的程式
ndarray.ndim	檢視陣列的維度	print(Grades.ndim)
ndarray.size	檢視陣列的元素數量	print(Grades.size)
ndarray.dtype	檢視陣列的元素型態	print(Grades.dtype)
ndarray.shape	檢視陣列的形狀	print(Grades.shape)

14 二維陣列操作

```
import numpy as np
Grades=np.array([[96,65,73],[88,76,82],[92,84,89],[82,73,64],[70,83,68]])
#取出某個學生的全部成績或某一科成績:
print(Grades[0])  # Grades[0]=Grades[0,:]=Grades[0,]
print(Grades[1])
print(Grades[0,0])  # Grades[0][0]= Grades[0,0]
print(Grades[1,2])
A=np.array([1,2,3,4,5,6,7,8,9,10,11,12]).reshape(4,3)
print(A.shape)
print(A.shape[0])
print(A.shape[1])
B=np.arange(15).reshape(3,5)
```

15 如何運用PYTHON產生N維陣列

- 直接輸入法
- 將串列轉為陣列
- 陣列的便捷產生方式
  - 運用range產生整數序列
  - 產生等差序列

16 運用PYTHON產生N維陣列

直接輸入法

```
import numpy as np
Weight=np.array([56, 72, 52, 66, 90])
print(type(Weight))
print(Weight.shape)
Array1=np.array([[31, 66, -45, 60, 6],[-38, 84, -24, 23, -2], [-86, 23, -43, 49, 56],[48, -49, 9, 37, -21]])
print(Array1)
print(Array1.shape)
```

將串列轉為陣列

```
import numpy as np
Weight_list= [56, 72, 52, 66, 90]
Weight_array=np.array(Weight_list)
Stock_array= np.array([[31, 66, -45, 60, 6],[-38, 84, -24, 23, -2], [-86, 23, -43, 49, 56],[48, -49, 9, 37, -21]]) print(Stock_array)
print(Stock_array.shape)
```

17 N維陣列轉換為二維陣列或一維陣列

```
Stock_list= [31, 66, -45, 60, 6,-38, 84, -24, 23, -2, -86, 23, -43, 49, 56,48, -49, 9, 37, -21]
Stock_array1=np.array(Stock_list)   #轉為一維陣列
Stock_array2= Stock_array1.reshape(4,5)    #轉為4行5列的二維陣列
print(Stock_array2)
Stock_array3= Stock_array1.ravel()  # 將二維陣列降維至一維陣列
print(Stock_array3)
```

18 陣列內的索引、切片及排序

```
Stock_ array= np.array([[31, 66, -45, 60, 6],[-38, 84, -24, 23, -2], [-86, 23, -43, 49, 56],[48, -49, 9, 37, -21]])

print(Stock_array[2:,1:4]) # 第3行到最後，第2列到第4列

print(np.sort(Stock_array,axis=0) # axis=0 表示按column對元素排序； axis=1 表示按row對元素排序
```

19 陣列內的相關運算

```
Stock_ array1= np.array([[31, 66, -45, 60, 6],[-38, 84, -24, 23, -2], [-86, 23, -43, 49, 56],[48, -49, 9, 37, -21]])

print(Stock_array1.sum(axis=0))

print(Stock_array1.sum(axis=1))

print(Stock_array1.sum())

print(Stock_array1.mean(axis=0))
```

```
print(Stock_array1.mean(axis=0))
print(Stock_array1.var(axis=0))  #變異數
print(Stock_array1.std(axis=0))   # 標準差
print(Stock_array1.min(axis=0))
print(Stock_array1.max(axis=0))
```

20 陣列內的相關運算練習題

- 計算矩陣Stock_ array1= np.array([[31, 66, -45, 60, 6],[-38, 84, -24, 23, -2], [-86, 23, -43, 49, 56],[48, -49, 9, 37, -21]])中，每一列(row)的平均值、標準差、最大值與最小值

21 陣列內的相關運算練習題_參考程式

- 計算矩陣Stock_ array2= np.array([[31, 66, -45, 60, 6],[-38, 84, -24, 23, -2], [-86, 23, -43, 49, 56],[48, -49, 9, 37, -21]])中，每一列(row)的平均值、標準差、最大值與最小值

```
import numpy as np
print(Stock_array2.mean(axis=1))
print(Stock_array2.var(axis=1))
print(Stock_array2.std(axis=1))
print(Stock_array2.min(axis=1))
print(Stock_array2.max(axis=1))
```

22 矩陣的基本運算

- 建立單位矩陣: I=np.eye(5)
- 兩個陣列加(+)減(-)乘(*)除(/)及次方(**): A1 and A2 需要相同的行數與列數
- Stock_ array2= np.array([[31, 66, -45, 60, 6],[-38, 84, -24, 23, -2], [-86, 23, -43, 49, 56],[48, -49, 9, 37, -21]])
- Coef_array=np.corrcoef(Stock_array2)

23 矩陣的基本運算(CONT.)

- np.diag(Stock_array2) # 矩陣的對角線
- np.triu(Stock_array2) # 矩陣上三角
- np.tril(Stock_array2) # 矩陣下三角
- np.transpose(Stock_array2) #矩陣的轉置
- Stock_array2.T # 矩陣的轉置

24 矩陣的基本運算

- import numpy.linalg as la
- la.dat(Stock_ array2) # 行列式
- la.inv(Stock_ array2) # 反矩陣
- la.svd(Stock_ array2) # 奇異值分解

25 廣播(BROADCAST)

- 因兩個陣列形狀必須相容才能進行運算，當形狀不同時，較小的陣列會依據NumPy提供的廣播機制擴張成較大的陣列相容的形狀，再進行運算

```
import numpy as np
A=np.array([9,8,7,6])
B=20
print(A+B)
C=np.array([[90,80,70,60],[10,20,30,40]])
print(A+C)
D=np.array([90,80,70])
print(A+D)
```

26 NUMPY官網的說明

- statistics: https://numpy.org/doc/stable/reference/routines.statistics.html