Taller Phyton: NumPy

Jorge Victorino, Miguel Barrero
Departamento de Ingeniería de Sistemas, C^2UC

2020

1. Introduction

NumPy es una potente librería enfocada a ciencias de computación bajo el lenguaje de programación **Phyton**.Esta librería le permite trabajar con matrices(Arrays) a nivel multidimensional,de forma fácil a partir de funciones que optimizan la manipulación de los datos. Con NumPy usted podrá de forma sofisticada (NumPy,2018):

- Crear matrices a nivel multidimensional.
- Realizar operaciones aritméticas entre matrices.
- Realizar indexación,iteración,extracción o corte(slicing) sobre los arrays.
- Utilizar herramientas para integrar código C/C++ y Fortran.

2. Arrays

Un array es una grilla de valores que contiene índices enteros positivos, al cual se le pueden asociar valores.Los arays pueden ser de n-dimensiones,definidos por un rango de valores(filas y columnas) que directamente expresan su dimensión.

Ahora usted podrá ver como se declaran arrays básicos,se accede a los datos y se visualizan algunas de sus características.

```
import numpy as np

a = np.array([1, 2, 3])          #Crea un array 1D
print(type(a))                   #Tipo de dato
print(a.shape)                   #Tamaño del array
print(a[0], a[1], a[2])          #Valores de los índices
a[0]= 5                          #Cambia el valor del elemento 0
```

```
print(a)                              #Array a
b = np.array([[1,2,3] , [4,5,6]])     #Array un array 2D
c = np.arange(1, 5)                   #Crea un array del 1 al 4
d = c.reshape(2, 2)                   #Redimensiona el array c
print(c)
print(d)
```

Tambien puede crear arrays con algunas funciones de Numpy e imprimirlos.

```
import numpy as np

a = np.zeros ((2,2))                  #Crea un array de ceros
b = np.ones ((1,2))                   #Crea un array de unos
c = np.full ((2,2))                   #Crea un array de una constante
d = np.eye ((2))                      #Crea una matriz de identidad
                                      #de 2x2
e = np.random.random((2,2))           #Crea un array lleno de
                                      #valores aleatorios
```

3. Arrays Indexing y Slicing

NumPy opera con indexación sobre los arrays. Esto le permite acceder a los datos contenidos a partir de un valor determinado. El acceso a los datos contenidos en el array, se realiza a través de un valor que se asocia directamente a una posición del mismo. Usted puede realizar un slicing (extraer) sobre el o los arrays que está trabajando (Johnson, 2018).

```
import numpy as np
nums = list(range())     # Crea una lista de enteros
print (nums)             # "[0, 1, 2, 3, 4]"
print (nums[2:4])        # Extrae del indice 2 al 4
                         # (excluyendolo)
print (nums[2:])         #Extrae desde el indice 2
print (nums[:2])         # Extrae del inicio hasta el
                         # indice 2(excluyendolo)
print (nums[:])          # ?
nums[2:4] = [8, 9]       # Asigna valores del 2 al 4
print (nums)             # "[0, 1, 8, 9, 4]"
```

Ahora inténtelo con una matriz de *m x n* .

```
import numpy as np

                                        #Crea un array (3,4)
                                        #[[1 2 3 4]]
                                        # [5 6 7 8]
                                        # [9 10 11 12]
a = np.array ([[1,2,3,4], [5,6,7,8], [9,10,11,12]])
                                        # Extraiga las dos primeras filas hasta la segunda
                                        # columna, generando una nueva matriz de (2, 2).
                                        # Imprimala.[[2 3][6 7]]
b = a[:2, 1:3]
print(a[0, 1])                          # Imprime 2
```

Para mayor entendimiento analice la siguiente figura

```
>>> a[0,3:5]
array([3,4])

>>> a[4:,4:]
array([[44, 45],
       [54, 55]])

>>> a[:,2]
array([2,12,22,32,42,52])

>>> a[2::2,::2]
array([[20,22,24]
       [40,42,44]])
```

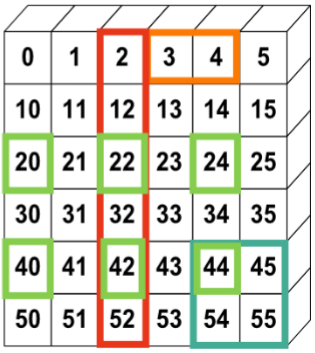

Figura 1: Avila, D. (2018). NumPy Cortes.[Figura]. Recuperado de: `http://damianavila.github.io/Python-Cientifico-HCC/3_NumPy.html`

Analice el siguiente indexado booleano en el cual se satisface una condición para extraer unos elementos

```
import numpy as np

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

bool_array = (a > 2)                    # Encuentra los elementos mayores
                                        # al valor 2 dentro del array y
                                        # devuelve verdadero o falso una vez
                                        # es evaluada la condición a un
                                        # nuevo array

print(bool_array)
# Imprime los valores que cumplen la condición de
```

```
# verdadero a partir del array original
print(a[bool_array])
```

4. Operaciones matemáticas

NumPy le permite realizar operaciones matemáticas entre arrays. Estas operaciones pueden ser utilizadas a partir de funciones propias de NumPy o como cualquier operación matemática.
Ejemplo

```
import numpy as np

# Crea dos arrays de tipo float
x = np. array ([[1 ,2] ,[3 ,4]], dtype=np. float64 )
y = np. array ([[5 ,6] ,[7 ,8]], dtype=np. float64 )
print(x.dtype)
print(y.dtype)

# Operación de suma
print(x + y)
print(np.add(x, y))

# Operación de resta
print(x - y)
print(np.subtract(x, y))

# Operación de multiplicación
print(x * y)
print(np.multiply(x, y))

# Operación de división
print(x / y)
print(np.divide(x, y))

# Raíz cuadrada
print(np.sqrt(x))
```

5. Ejercicios

Para los siguientes ejercicios utilice la librería NumPy.

- Cree una matriz m de 5 x 5 partiendo de un array unidimensional entre valores enteros negativos y positivos, estos no deben repetirse.

- Multiplique y sume a la matriz m un escalar entero positivo. Imprima de forma directa la matriz y cree una estructura de control cíclica que la imprima.
- Cree una nueva matriz n a partir de la matriz m e imprima los valores que satisfacen la condición x >15.
- Realice un slicing para una nueva matriz y que contenga los datos de las filas 2 y 4 entre las columnas 1 y 3 de la matriz m.
- A partir de una estructura de control condicional y cíclica determine si los elementos de la matriz y son positivos.

Referencias

Johnson, J. (2018). Python numpy tutorial.
NumPy, d. (2018). Numpy.