Unidad 04
Python para computación científica

Curso de geoprocesamiento de datos con Python
2016

Cayetano Benavent Viñuales
Analista GIS en Geographica
cayetano.benavent@geographica.gs

Unidad 04 - Sumario de contenidos

¿Qué es NumPy?

"NumPy is the fundamental package for scientific computing with Python."

http://www.numpy.org

La práctica totalidad del software que realiza cómputos intensivos en Python, utiliza Numpy. Sin NumPy, el lenguaje Python no sería lo que es hoy en el mundo de la computación científica.

¿Qué es NumPy?

Es importante recalcar que NumPy, a pesar de su uso masivo, no viene preinstalado con Python.

Si queremos usar Numpy, debemos instalarlo.

Es una librería independiente, perteneciente a lo que se conoce como The SciPy Stack.

¿Qué es The SciPy Stack?

"SciPy is a Python-based ecosystem of open-source software for mathematics, science, and engineering.

In particular, these are some of the core packages:"

NumPy
Base N-dimensional array package

SciPy library
Fundamental library for scientific computing

Matplotlib
Comprehensive 2D Plotting

IP[y]: IPython

IPython
Enhanced Interactive Console

Sympy
Symbolic mathematics

pandas
Data structures & analysis

http://scipy.org/

Principales características de NumPy

“ - a powerful N-dimensional array object

- sophisticated (broadcasting) functions

- tools for integrating C/C++ and Fortran code

- useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data.

(...)”

http://www.numpy.org

Importación de NumPy

La convención para importar la librería NumPy, según la documentación oficial, es:

```
>> import numpy as np
```

Ello hará que nuestro código sea más legible y coherente e integrable en otras aplicaciones.

NumPy arrays

El array NumPy (`ndarray`) es el principal tipo de objeto manejado por la librería.

Es un array N-dimensional.

Son mucho más eficientes y rápidos que otros tipos de objetos en Python, especialmente manejando datos de tipo numérico.

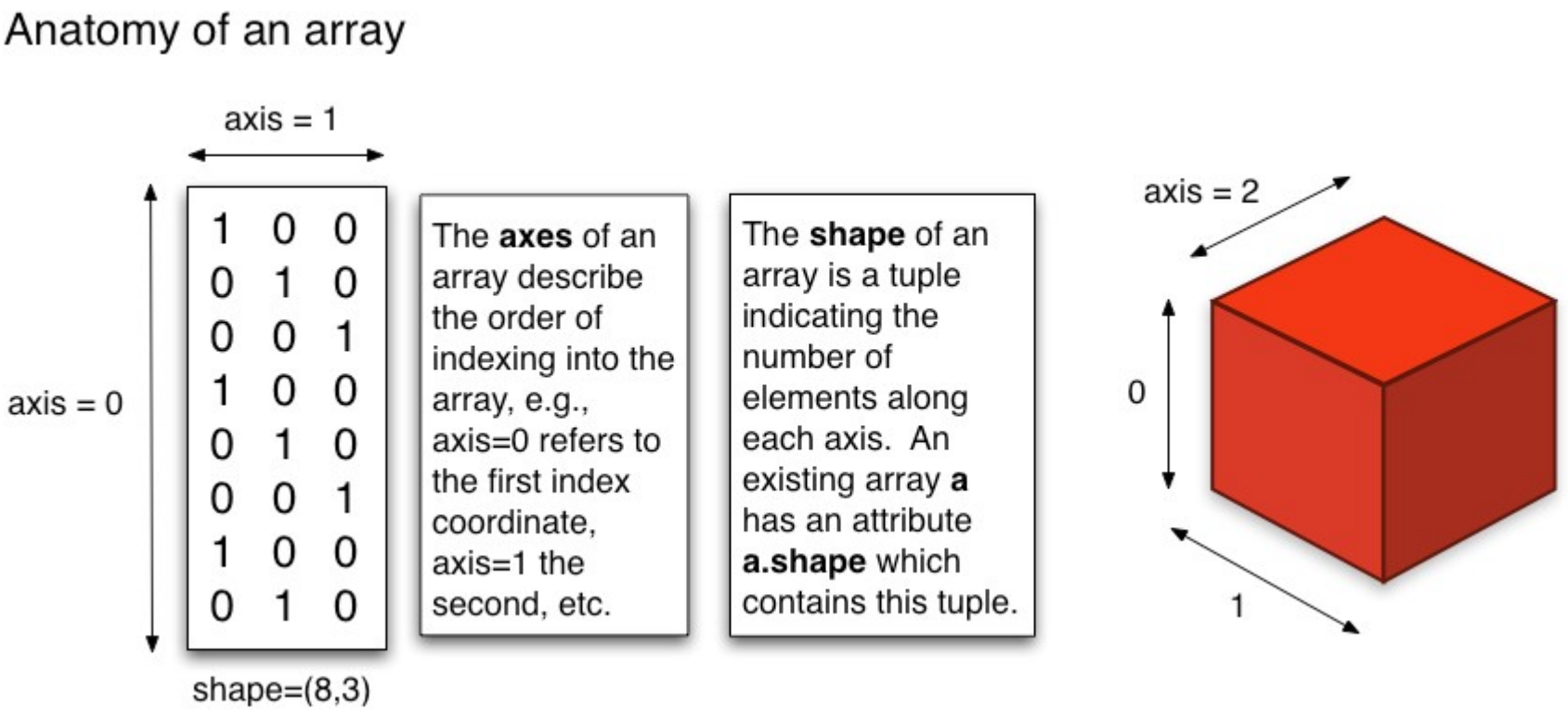
De hecho, el array NumPy está diseñado específicamente para computación científica.

NumPy arrays

Un array NumPy es homogéneo en cuanto al tipo de dato que almacena.

Un array NumPy es, por tanto, una colección de elementos del mismo tipo. Ejemplo: un array de integers, un array de floats, un array de strings, etc.

Esta limitación es precisamente una de las características que lo hacen tan eficiente.



Fuente: http://pages.physics.cornell.edu/~myers/teaching/ComputationalMethods/python/anatomyarray.png

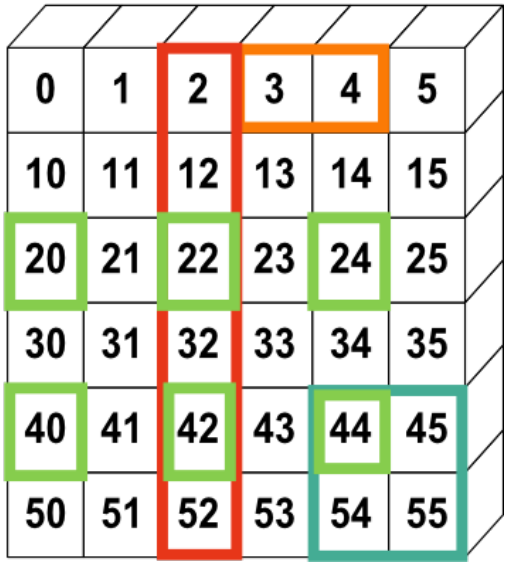
Slicing NumPy arrays

```
>>> 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]])
```



Fuente: http://www.scipy-lectures.org/_images/numpy_indexing.png

Manejo de NumPy arrays

Para entender mejor como manejar NumPy arrays, realizaremos un ejercicio práctico siguiendo el Notebook unit04_01.

Operaciones “element wise”

Este tipo de operaciones son las más comunes, y supone el distribuir una operación a todo el array.

Ejemplo de una multiplicación “element wise”:

```
>>> a = np.array([1, 2, 3, 4])
>>> a * 2
    array([2, 4, 6, 8])
```

Operaciones con NumPy arrays

CUIDADO: Hay que tener en cuenta que multiplicar dos arrays no efectúa una multiplicación de matrices.

Para obtener el producto matricial hay que usar el método `dot`.

```
>> b.dot(c)
```

Operaciones de reducción

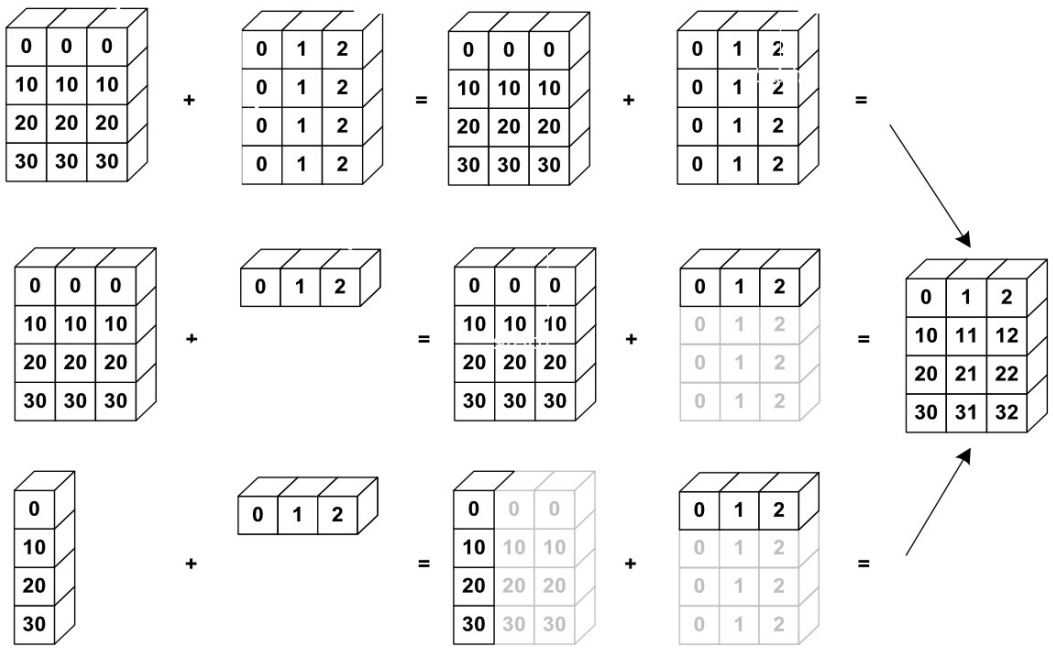
Operaciones con el array completo o con dimensiones completas.

Ejemplos: sum, max, min, mean, std, etc.

```
>>> a.sum()
>>> a.max()
>>> a.min()
>>> a.mean()
```

Operaciones de broadcasting

Operaciones entre arrays de diferentes tamaños.



Fuente: http://www.scipy-lectures.org/_images/numpy_broadcasting.png

Operaciones con NumPy arrays

Para entender mejor como realizar operaciones con NumPy, realizaremos un ejercicio práctico siguiendo el Notebook unit04_02.

Matplotlib es la librería gráfica (2D) más importante de Python. La web del proyecto es:

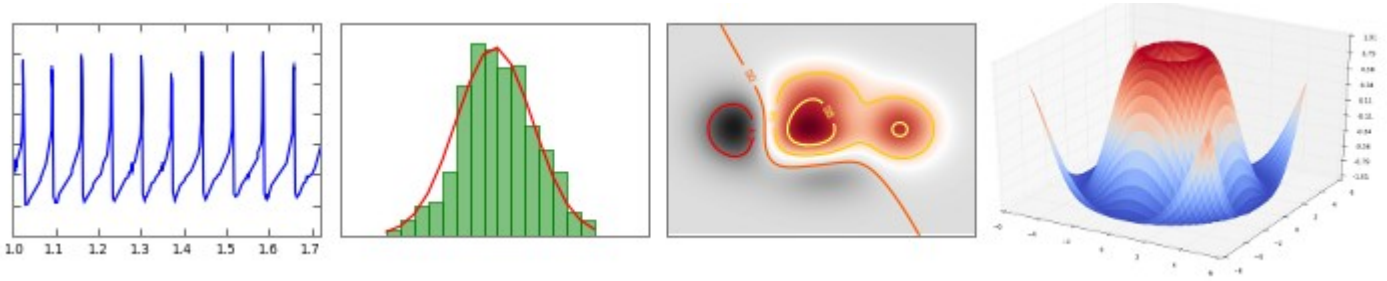
http://matplotlib.org/

Las posibilidades de uso son muy amplias:

“ matplotlib can be used in python scripts, the python and ipython shell (ala MATLAB® or Mathematica®), web application servers, and six graphical user interface toolkits.”

Matplotlib es una librería MUY grande. Llegar a conocerla en profundidad requiere muchísimo tiempo.

La cantidad de soluciones gráficas que ofrece es realmente sorprendente.



Para entender mejor como generar gráficos con Matplotlib, realizaremos un ejercicio práctico siguiendo el Notebook unit04_03.

matplotlib

Para afianzar la compresión de la librería Matplotlib, veremos un ejemplo práctico siguiendo el Notebook unit04_04.

Un último ejercicio práctico con Matplotlib, veremos un ejemplo para generar histogramas siguiendo el Notebook unit04_05.

Análisis y manipulación de datos: Introducción a Pandas

Pandas es una librería construida sobre NumPy especializada en análisis y manipulación de datos:

http://pandas.pydata.org/

Incorpora muchas de las “buenas ideas” del lenguaje R al mundo de Python.

Según sus autores:

“It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis / manipulation tool available in any language. It is already well on its way toward this goal.”

Análisis y manipulación de datos: Introducción a Pandas

Algunas de sus capacidades:

- *Easy handling of missing data (represented as NaN).*
- *Size mutability: columns can be inserted and deleted from DataFrame.*
- *Automatic and explicit data alignment: objects can be explicitly aligned to a set of labels.*
- *Powerful, flexible group by functionality to perform split-apply-combine operations on data sets.*
- *Make it easy to convert ragged, differently-indexed data in other Python and NumPy data structures into DataFrame objects*
- *Intelligent label-based slicing, fancy indexing, and subsetting of large data sets*
- *Intuitive merging and joining data sets*
- *Flexible reshaping and pivoting of data sets*
- *Hierarchical labeling of axes.*
- *Robust IO tools for loading data from a lot of formats (CSV, Excel, databases, HDF5).*
- *Time series-specific functionality."*

Análisis y manipulación de datos: Introducción a Pandas

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

Para entender mejor como funciona Pandas, realizaremos un ejercicio práctico siguiendo el Notebook unit04_06.

Análisis y manipulación de datos: Introducción a Pandas

Para profundizar en la comprensión de la librería Pandas, realizaremos un ejercicio práctico siguiendo el Notebook unit04_07.

Algoritmos avanzados: introducción a Scipy

“SciPy (pronounced “Sigh Pie”) is open-source software for mathematics, science, and engineering..”

https://www.scipy.org/scipylib/

SciPy utiliza como elemento de trabajo base el array NumPy.

La velocidad y potencia de la librería se debe a que SciPy está construido sobre el robusto repositorio de algoritmos Netlib (mayormente escrito en C y Fortran):

http://www.netlib.org/

Algoritmos avanzados: introducción a Scipy

SciPy se organiza en subpackages, cubriendo diferentes dominios de computación científica

Subpackage	Description
cluster	Clustering algorithms
constants	Physical and mathematical constants
fftpack	Fast Fourier Transform routines
integrate	Integration and ordinary differential equation solvers
interpolate	Interpolation and smoothing splines
io	Input and Output
linalg	Linear algebra
ndimage	N-dimensional image processing
odr	Orthogonal distance regression
optimize	Optimization and root-finding routines
signal	Signal processing
sparse	Sparse matrices and associated routines
spatial	Spatial data structures and algorithms
special	Special functions
stats	Statistical distributions and functions
weave	C/C++ integration

Algoritmos avanzados: introducción a Scipy

Para introducirnos en el funcionamiento de la librería SciPy, realizaremos un ejercicio práctico siguiendo el Notebook unit04_08.

Bibliografía

Selección bibliográfica:

- Oliphant, Travis E. (2015): “Guide to NumPy: 2nd Edition”. Continuum Press.

- Madhvan, Samir (2015): “Mastering Python for Data Science”. Packt Publishing.

- McKinney, Wes (2012): “Python for Data Analysis. Data Wrangling with Pandas, NumPy, and IPython”. O'Reilly.

- Bressert, Eli (2012): “SciPy and NumPy: An Overview for Developers”. O'Reilly.