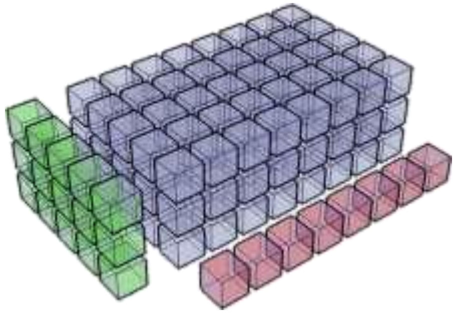
Python-Numpy

Agenda

- Introduction
- History, usage
- Universal Functions
- Indexing, Slicing and Iterating
- Stacking -splitting arrays
- Broadcasting
- Reading from csv files

NumPy

Numerical Python

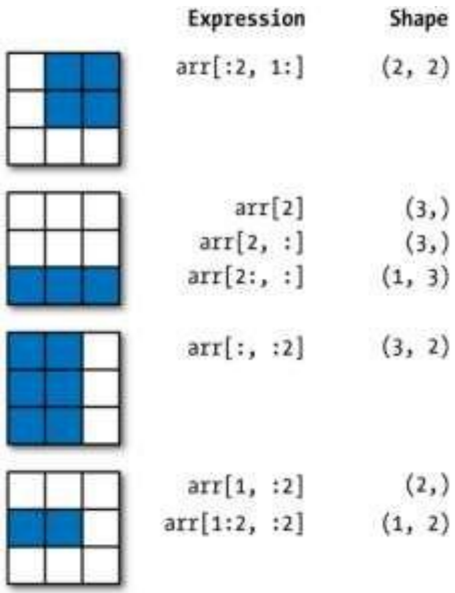


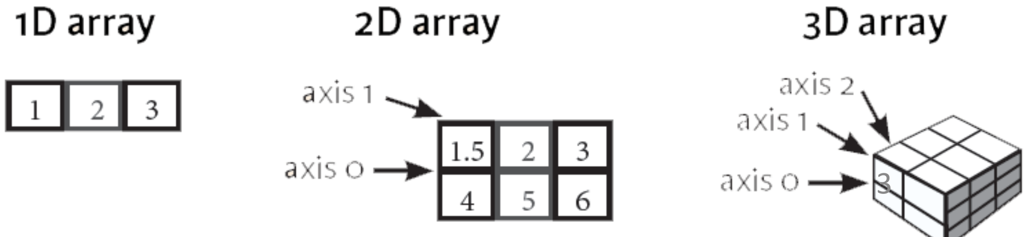
astropy
BIOPYTHON
DIPY
NIPY
SunPy
(and many, many more)
PyTables
NetworkX
SM StatsModels Statistics in Python
scikits-image image processing in python
learn
matplotlib
pandas
PyMC
SciPy
NumPy
Cython
SymPy
IP[y]: IPython
python
jupyter

Introduction

- Numerical Python (Numpy) has greater role for numerical computing in Python.
- It provides the data structures, algorithms, and library glue needed for most scientific applications involving numerical data in Python.
- It has fast and efficient multidimensional

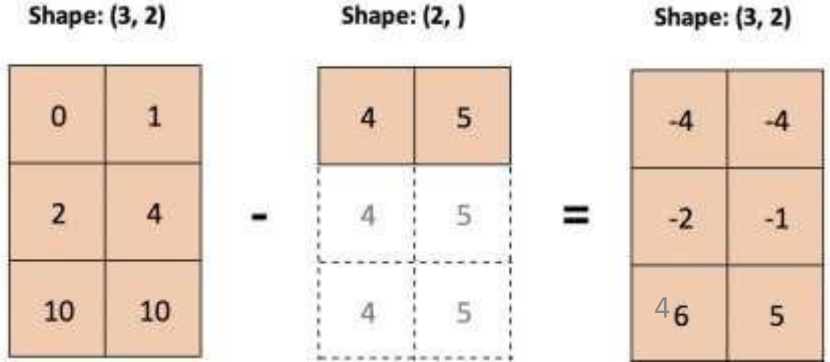
  (N-dimensional) array object ndarray
- Functions for performing element-wise computations with arrays or mathematical operations between arrays
- It has tools for reading and writing array-based datasets to disk





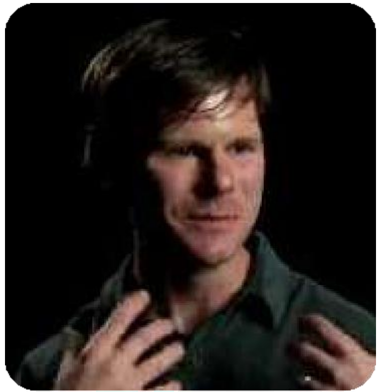
Introduction

- It is useful for Linear algebra operations, Fourier transform, and random number generation
- It has sophisticated (broadcasting) functions
- It has tools for integrating C/C++ and Fortran code
- It provides an efficient interface to store and operate on dense data buffers
- NumPy arrays form the core of nearly the entire ecosystem of data science tools in Python
- Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.
- NumPy is licensed under the BSD license



History

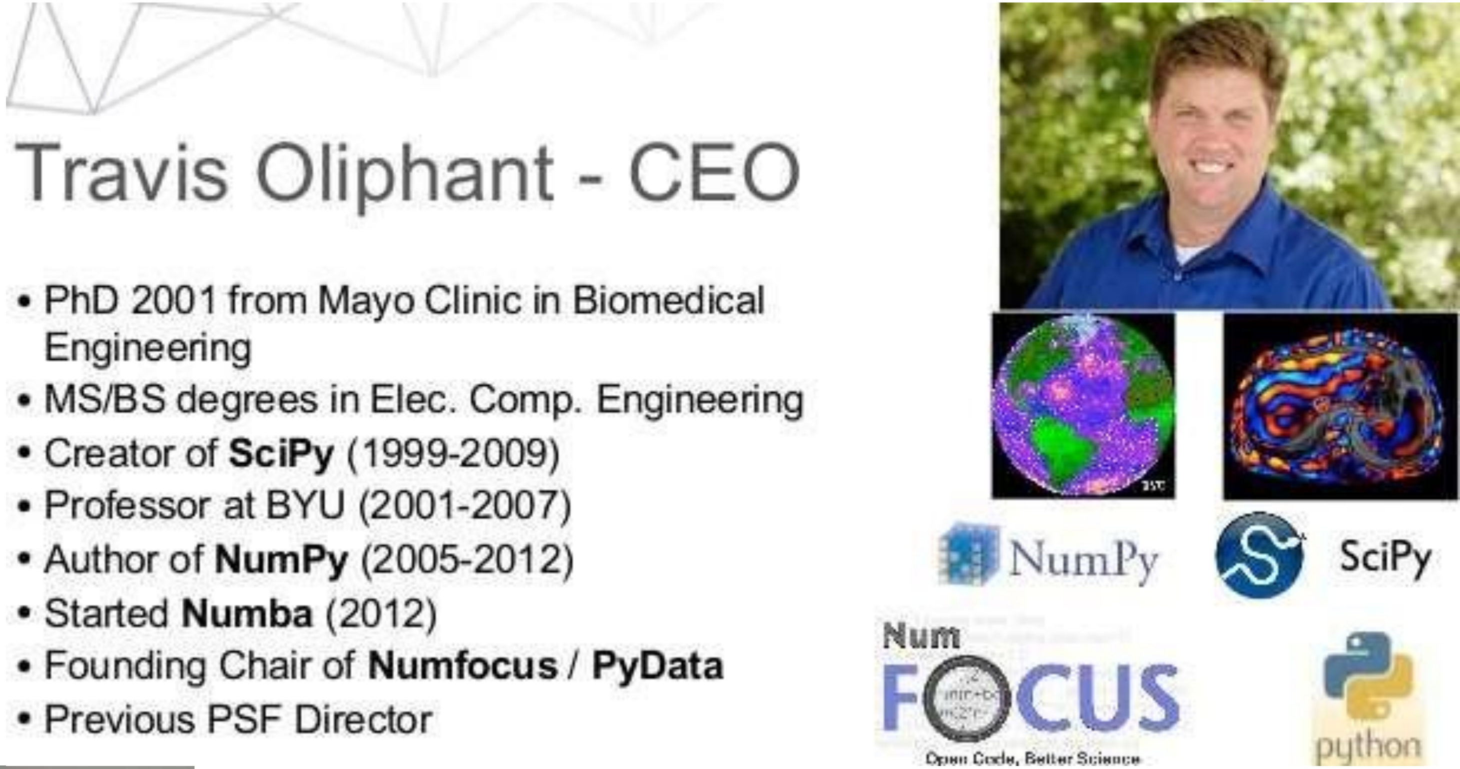
- **NumPy** derives from an old library called Numeric, which was the first array object built for Python. (written in 2005 launched in 2006)
- Numeric was quite successful and was used in a variety of applications before being phased out.



Jim Hugunin

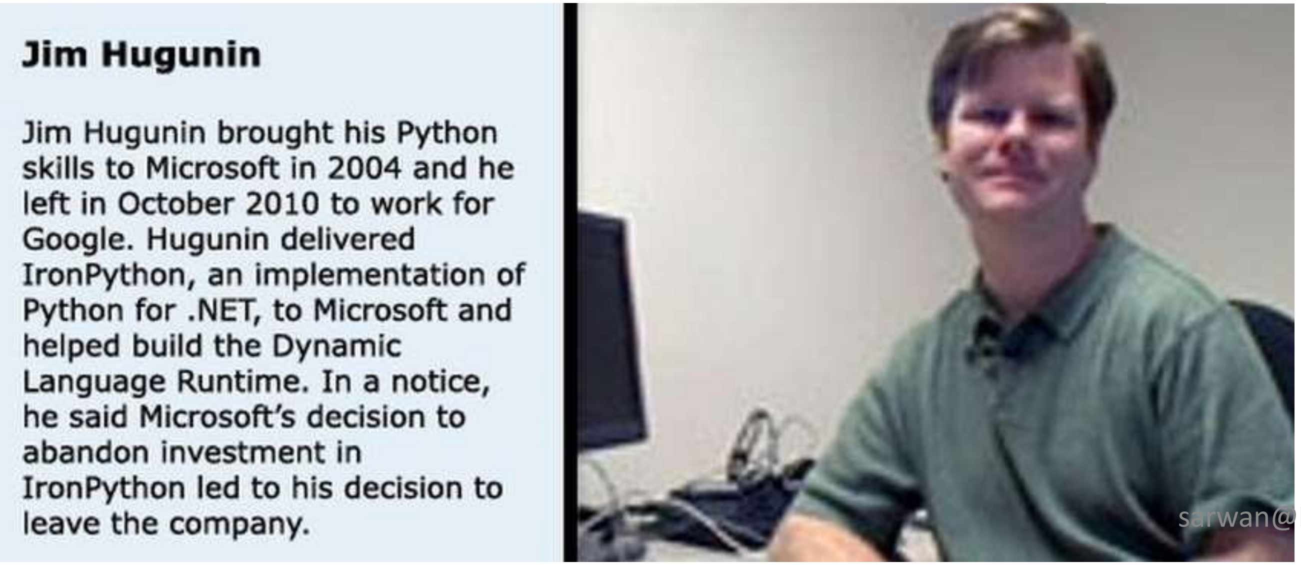
Jim Fulton

Person	Package	Year
Jim Fulton	Matrix Object in Python	1994
Jim Hugunin	Numeric	1995
Perry greenfield, Rick white, Todd Miller	Numarray	2001
Travis Olipant	NumPy	2005



Travis Oliphant - CEO

- PhD 2001 from Mayo Clinic in Biomedical Engineering
- MS/BS degrees in Elec. Comp. Engineering
- Creator of **SciPy** (1999-2009)
- Professor at BYU (2001-2007)
- Author of **NumPy** (2005-2012)
- Started **Numba** (2012)
- Founding Chair of **Numfocus** / **PyData**
- Previous PSF Director



Jim Hugunin

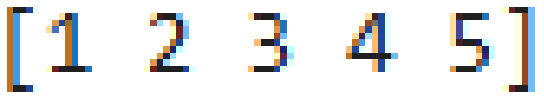
Jim Hugunin brought his Python skills to Microsoft in 2004 and he left in October 2010 to work for Google. Hugunin delivered IronPython, an implementation of Python for .NET, to Microsoft and helped build the Dynamic Language Runtime. In a notice, he said Microsoft's decision to abandon investment in IronPython led to his decision to leave the company.

- NumPy’s main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers.
- In NumPy dimensions are called *axes*.
- NumPy defines N-dimensional array type called ndarray(also known by the alias array). It describes the collection of items of the same type. Items in the collection can be accessed using a zero-based index.
- numpy.array is not the same as the Standard Python Library class array.array, which only handles one-dimensional arrays.

- Creating Arrays

Basic Arrays

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
print(arr)
```

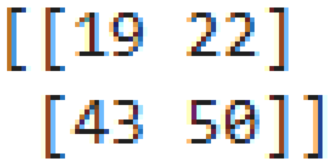
Output: 

```
[1 2 3 4 5]
```

- **Matrix Operations**

Arithmetic operations and Matrix Multiplication

```
mat1 = np.array([[1, 2], [3, 4]])
mat2 = np.array([[5, 6], [7, 8]])
dot_product = np.dot(mat1, mat2)
print(dot_product)
```

Output: 

```
[[19 22]
 [43 50]]
```

- **Indexing and Slicing:**

Simple Indexing and Slicing

```
arr = np.array([10, 20, 30, 40])
print(arr[1:3])  # Output: [20, 30]
```

arr[1:3] slices the array from index 1 to 3, but it excludes the element at index 3 because slicing in Python (and NumPy) follows the rule that the **end index is exclusive**.

So, this will include elements at index 1 and 2, which are 20 and 30, respectively.

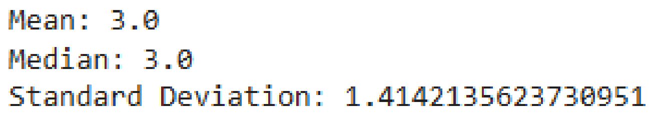
- **Statistical Functions:**

Mean, Median, and Standard Deviation

```
arr = np.array([1, 2, 3, 4, 5])
mean = np.mean(arr)
median = np.median(arr)
std_dev = np.std(arr)
print("Mean:", mean)
print("Median:", median)
print("Standard Deviation:", std_dev)
```

np.mean(arr) calculates the mean (average) of the array. The mean is the sum of all elements divided by the number of elements. **15/5 = 3**
np.median(arr) calculates the median of the array. The median is the middle value when the elements are sorted in order. If there’s an odd number of elements, it’s the middle element. Since the array is already sorted ([1, 2, 3, 4, 5]), the median is the middle value, which is 3.So, **median = 3.0**
The standard deviation is a measure of how spread out the values in the dataset are around the mean. For the array [1, 2, 3, 4, 5], the values are spread out moderately, so the standard deviation comes out to **approximately 1.414.**

Output:

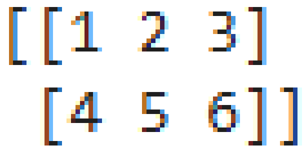


```
Mean: 3.0
Median: 3.0
Standard Deviation: 1.4142135623730951
```

- **Reshape an Array**

```
matrix = np.array([1, 2, 3, 4, 5, 6])
reshaped = matrix.reshape(2, 3)
print(reshaped)
```

reshape(2, 3) rearranges the original 1D array into a 2D matrix with 2 rows and 3 columns.This is a very useful operation when you want to change the shape of data to make it compatible with various mathematical or machine learning operations.

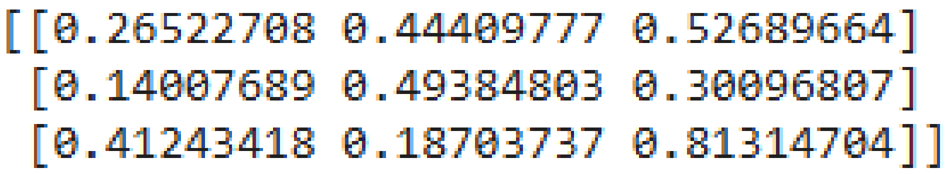


```
[[1 2 3]
 [4 5 6]]
```

- **Random Numbers**

```
#Generating Random Numbers
rand_array = np.random.rand(3, 3)
print(rand_array)
```

The numbers in the output will vary each time you run the code because they are randomly generated. Here’s an example of what the output might look like:



```
[[0.26522708 0.44409777 0.52689664]
 [0.14007689 0.49384803 0.30096807]
 [0.41243418 0.18703737 0.81314704]]
```

Check out this colab file for more:

https://github.com/lovnishverma/Python-Getting-Started/blob/main/NumPY.ipynb

https://colab.research.google.com/github/lovnishverma/Python-Getting-Started/blob/main/NumPY.ipynb