Self Introduction

Master of Molecular Science and Software Engineering
University of California, Berkeley
(2023-2025)

Honours Bachelor of Science
Chemical Physics specialist, Mathematics minor
University of Toronto
(2019-2023)

Teaching Experience

Geniebook

- Science Teacher
  Geniebook Vietnam
  (2023-2024)

UNIVERSITY OF TORONTO

- Teaching Assistant & Peer Tutor
  University of Toronto
  (2021-2023)

Highlights of Research Experience

UNIVERSITY OF TORONTO

- Conducted research on heat dissipation and cooling in nanoelectronic junctions of molecular devices using MATLAB under the supervision of Professor Dvira Segal.

UCSF Bakar Computational Health Sciences Institute

- Analyzed 4M+ patient records using Python tools on Wynton HPC.
- Processed large-scale EHRs.
- Built models for cancer subtype and readmission prediction.
- Visualized patterns with Seaborn and Matplotlib.

Introduction to NumPy

Programming in Python

Joy (Như) Ha
joyha@berkeley.edu
Office hours:
.....
Github

01 - Introduction

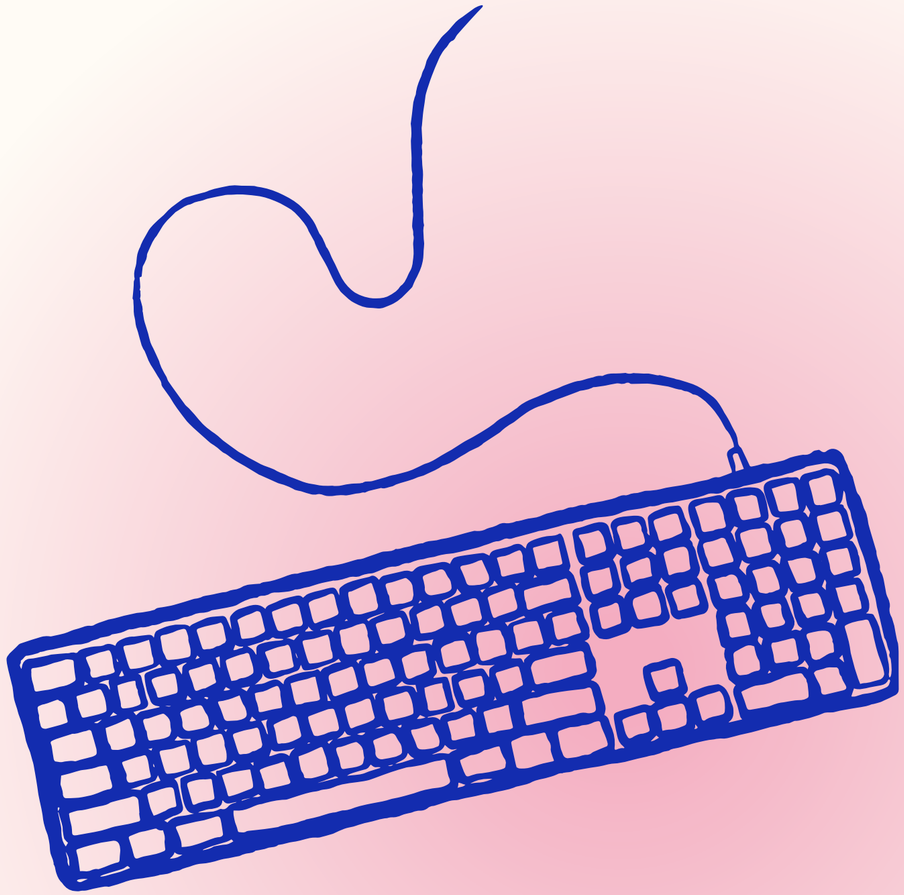
02 - Arrays

03 - Operations on Arrays

04 - Resources

Good to know

- Git
- Jupyter Notebook
- Python data structures (List, Tuple, ...)



01 - Introduction

A fundamental package for scientific computing in Python.

A Python library that provides a **HOMOGENEOUS MULTIDIMENSIONAL ARRAY** object, various derived objects (such as masked arrays and matrices), and an assortment of routines for **FAST** operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, basic linear algebra, basic statistical operations, random simulation and much more.

SMALLER memory use than LIST - Python built-in data structure)

```
import numpy as np
```

01 - Introduction

NumPy = ? + ?

- Aimed to build a more capable, open-source numerical toolkit using Python.
- Rooted in openness—relying on collaboration and community to grow and sustain these tools.

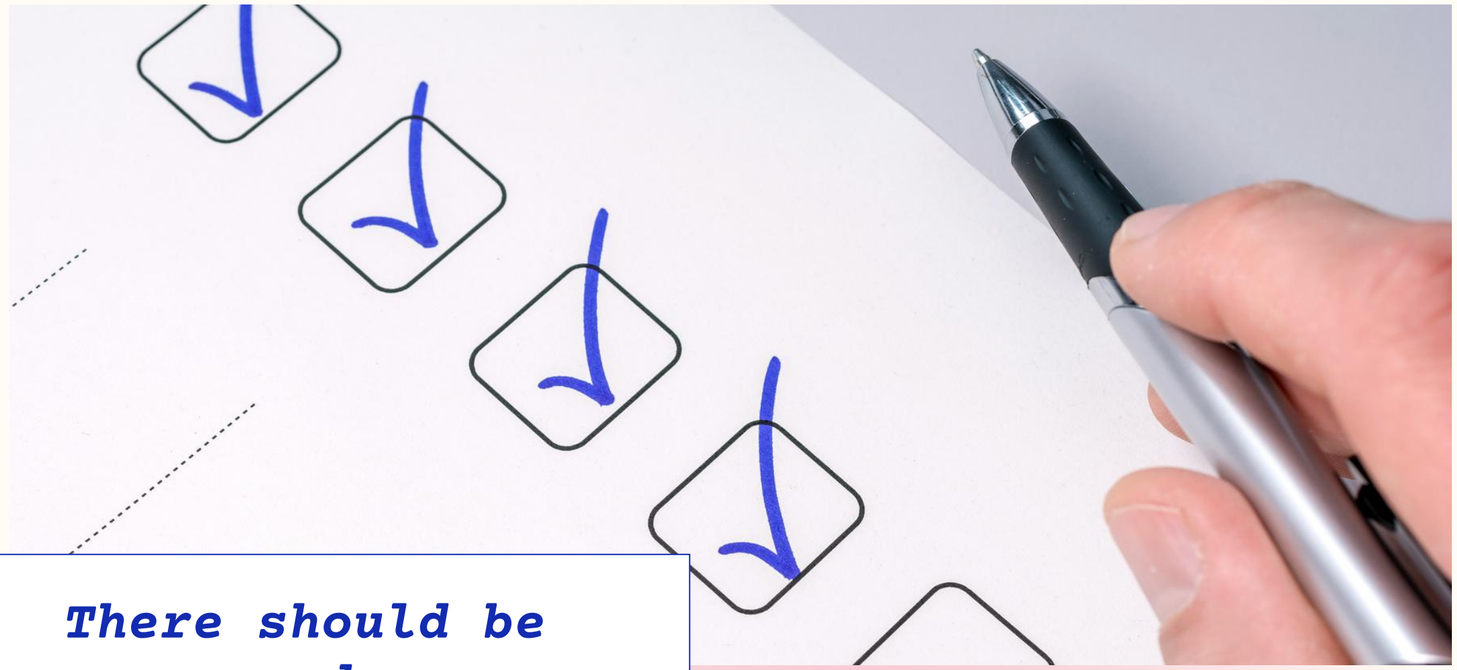
Numeric and Numarray

Unification of Numeric and Numarray

NumPy 1.0

Widely downloaded and foundational in data science and engineering

01 - Introduction



There should be one-- and preferably only one --obvious way to do it.

From the Zen of Python

This philosophy is often not applied in NumPy.

Example:

transposing an array

```
# Property
a2d.T
# Method
a2d.transpose()
# Function
np.transpose(a2d)
```

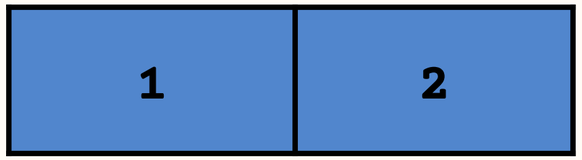
02 - Arrays

`ndarray` = N-dimensional array

Core of NumPy

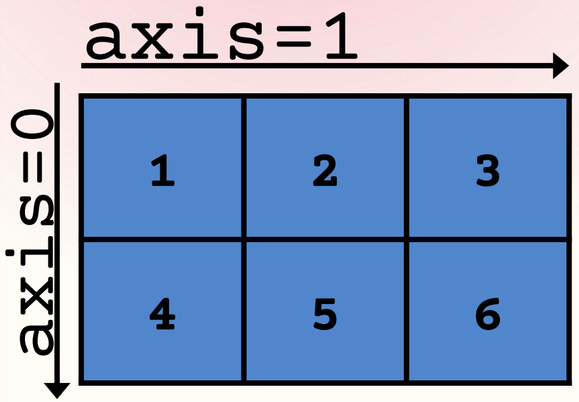
02.01 - Array Creation

Creating 1D array




```
>>> array_
>>> array_
array([1,
```

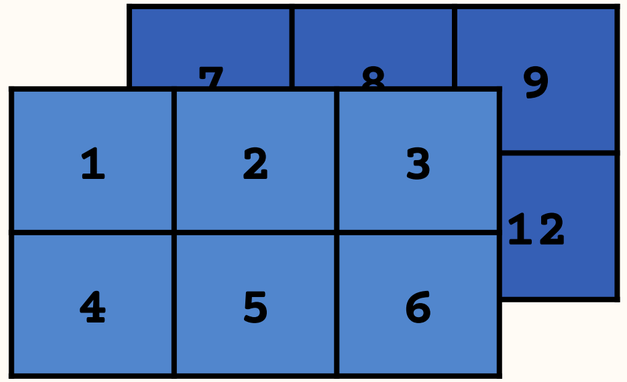
Creating 2D array




```
6]])
```

Other functions:
np.zeros(), np.ones(), np.empty(), np.arange(), np.linspace()

Creating 3D array



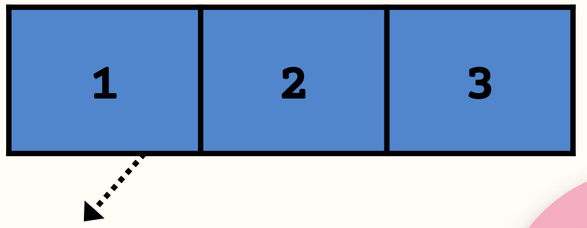

```
>>> array_3D = np.array([[[1, 2, 3], [4, 5, 6]],
[[7, 8, 9], [10, 11, 12]]])
>>> array_3D
array([[[ 1,  2,  3],
        [ 4,  5,  6]],

       [[ 7,  8,  9],
        [10, 11, 12]]])
```

Read more about how 3D arrays are used in image processing

02.02 - Selecting Array Entries

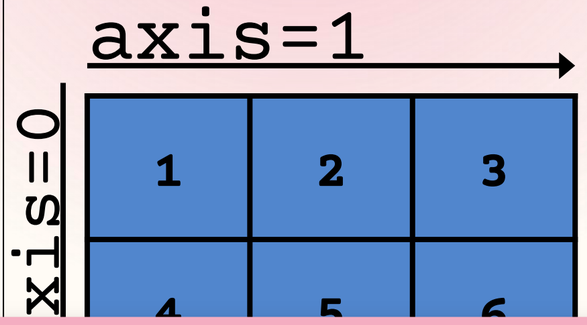
1D array



array_1D[0]

```
>>> data[1]
2
>>> data[0:2]
array([1, 2])
>>> data[1:]
array([2, 3])
>>> data[-2:]
array([2, 3])
```

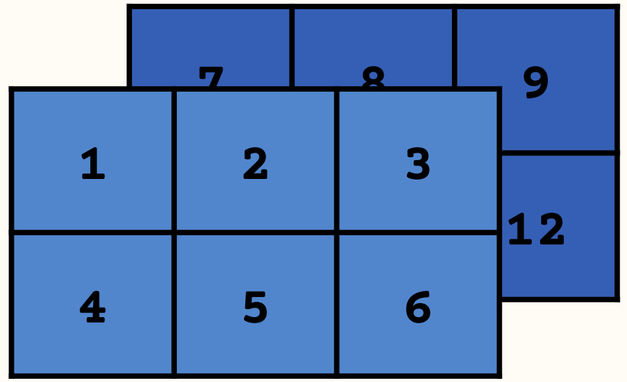
2D array




```
array([4, 5, 6])
>>> array_2D[:,0]
array([1, 4])
>>> array_2D[0,:]
array([1, 2, 3])
```

How to find the axes of an N-dimensional NumPy array?

3D array




```
>>> array_3D[0,0,1]
2
>>> array_3D[0]
array([[1, 2, 3],
       [4, 5, 6]])
>>> array_3D[0,1]
array([4, 5, 6])
>>> array_3D[1,:,0]
array([ 7, 10])
```

02.03 - NumPy Arrays versus Python Lists

NumPy arrays	Python lists
• Homogeneous is a key factor • Elements of an array are stored contiguously in memory • Element-wise operation is available • Faster numerical operations • ...	• Homogeneous or Heterogeneous • Elements of a list need not be contiguous in memory • Element-wise operation is not available • Slower numerical operations • ...

When to use NumPy arrays?

02.04 - Data Types

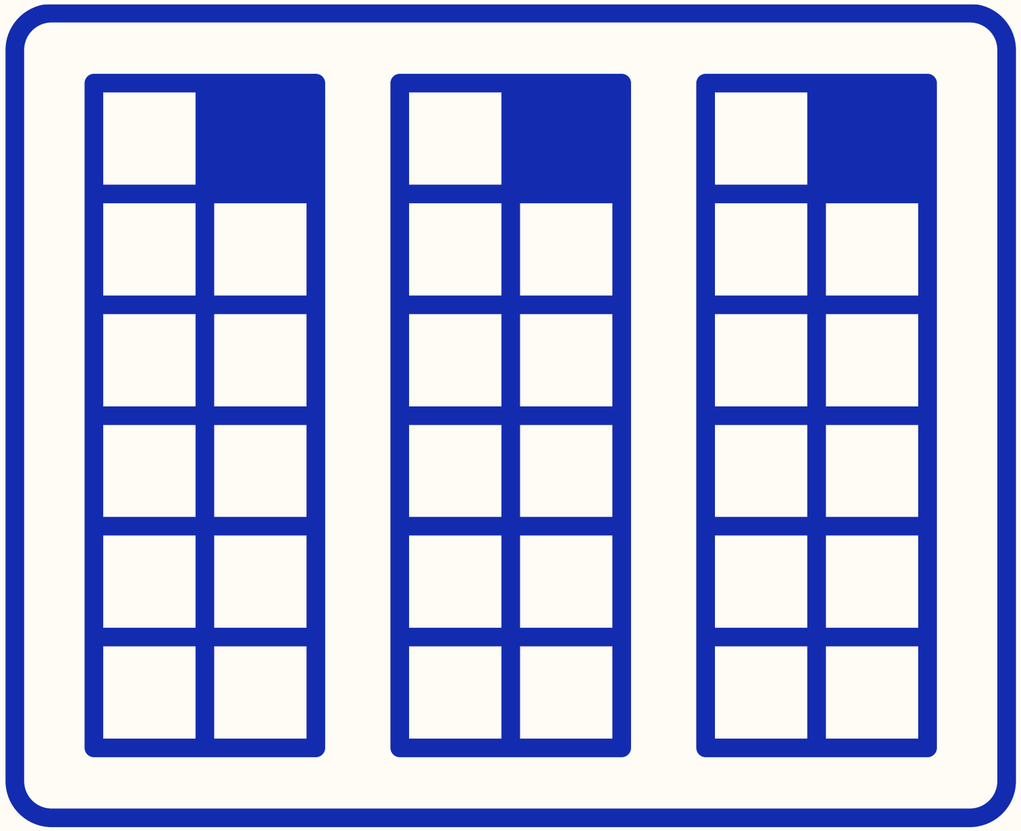
Homogeneity: all elements within a standard ndarray must be of the same data type.

- **i - integer**
- **b - boolean**
- **u - unsigned integer**
- **f - float**
- **c - complex float**
- m - timedelta
- M - datetime
- O - object
- S - string
- U - unicode string
- V - fixed chunk of memory for other type (void)

```
>>> a = np.array([1,2,3])
>>> a.dtype
dtype('int64')
>>> b = np.array([1.0,2.0,3.5])
>>> b.dtype
dtype('float64')
>>> c = np.array([1, 2.5])
>>> c.dtype
dtype('float64')
```

02.05 - Common Array Attributes

Attribute	Description
ndim	returns number of dimension of the array
size	returns number of elements in the array
dtype	returns data type of elements in the array
shape	returns the size of the array in each dimension



03 - Operations on Arrays

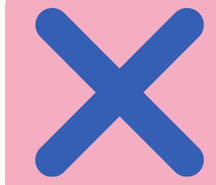
03.01 - Basic array operations

ELEMENT-WISE

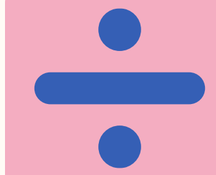
```
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])
```

```
>>> a + b
array([5, 7, 9])
```

```
>>> b - a
array([3, 3, 3])
```



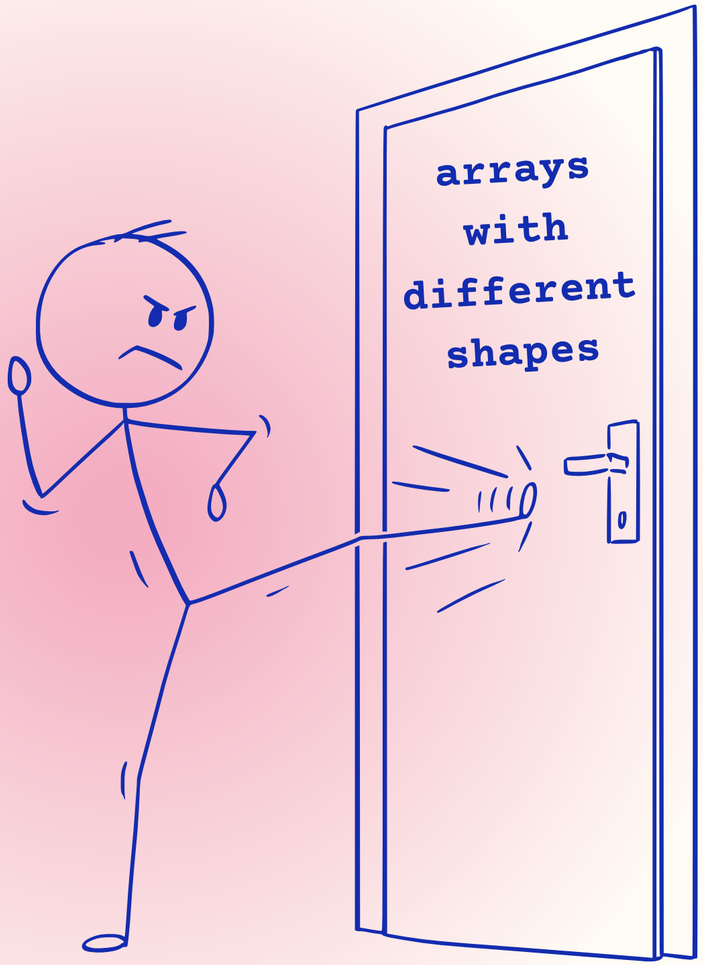
```
>>> a * b
array([4, 10, 18])
```



```
>>> a / b
array([0.25, 0.4 , 0.5 ])
```

More ...

03.02 - Broadcasting



- Broadcasting in NumPy enables arithmetic operations on arrays of different shapes without reshaping.
- It adjusts the smaller array to match the larger one by replicating values.
- This efficiency reduces memory usage and eliminates the need for loops.

```
>>> a = np.array([1, 2, 3])
>>> a + 1
array([2, 3, 4])
>>> b = np.array([[1, 3, 5], [7, 9, 11]])
>>> a + b
array([[3,6,9],
       [9,12,15]])
```

Mr. Broadcasting

Array dimensions must be compatible—either equal or one must be 1. Otherwise, a ValueError occurs.

03.03 - More useful operations

- Aggregation: a.max(), a.min(), a.sum(), a.mean(), a.median(), a.std(),...
- Transposing and reshaping: array_2D.reshape(3,2), array_2D.T or array_2D.transpose()
- Reversing: np.flip(a)

Practice this demo notebook to better understand NumPy library Open

04 - Resources

NumPy Installation

https://numpy.org/install/

NumPy User Guide

https://numpy.org/doc/stable/user/absolute_beginners.html

NumPy Source Code

https://github.com/numpy/numpy

NumPy Tutorial

https://www.w3schools.com/python/numpy/

More ...

Thank you!

Q&A Time

Joy Ha
joyha@berkeley.edu
Click here to schedule an appointment