

LECTURE 22

NUMPY

MCS 275 Spring 2023

David Dumas

LECTURE 22: NUMPY

Reminders and announcements:

- [Project 3](#) available; due 6pm on Fri Mar 17.
- Install Pillow now (if possible) to make lab this week smoother.
- I will be out of work on Fri Mar 10, and will post an asynchronous lecture video in place of our usual meeting that day.

A GOOD BOOK

For numpy, matplotlib, and a few other topics from MCS 275, I strongly recommend reading:

- *Python Data Science Handbook* by Jake VanderPlas

It is available for free online. Chapter 2 is about numpy.

INSTALLING NUMPY

In most cases, pip is all you need:

```
python3 -m pip install numpy
```

Other methods are described in the Numpy docs.

Test:

```
>>> import numpy
>>> numpy.__version__
'1.17.4'
```

IMPORT AS

You can give a module a new name at import time, e.g.

```
import math as sun
sun.tan(0.5)
```

Since numpy has a lot of global names, most people import it under a shorter name to save typing:

```
import numpy as np
```

NUMPY PURPOSE

- Fast, type-homogeneous, multidimensional arrays
 - e.g. vector, matrix, tensor, ...
- Large library of mathematical functions and algorithms (especially linear algebra)

Numpy is one of the most-used Python packages in scientific computing (computational math, data science, machine learning, ...).

ARRAYS

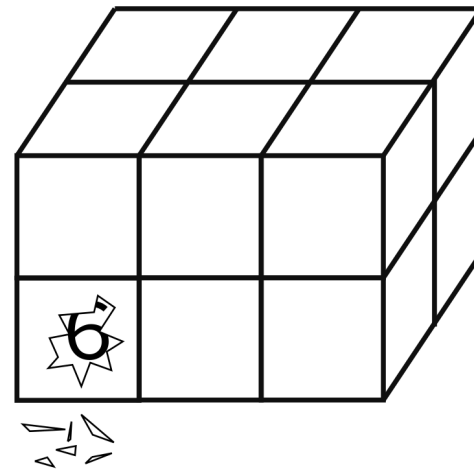
1-dimensional array of shape (7,)

2	7	5	2	0	2	1
---	---	---	---	---	---	---

2-dimensional array of shape (2,4)

2	7	5	0
4	8	1	1

3-dimensional array of shape (2,2,3)



Implemented in `np.ndarray` class, usually made with `np.array`.

Without numpy:

```
v = [2,3]
w = [3,-2]
v + w      # [2,3,3,-2]
3*v        # [2,3,2,3,2,3]
v.dot(w)   # fail!
A = [ [2,1], [1,1] ]
type(A)    # list
A*v        # fail!
```


With numpy:

```
v = np.array([2,3])
w = np.array([3,-2])
v + w      # [5,1]
3*v        # [6,9]
v.dot(w)   # 0
A = np.array([ [2,1], [1,1] ])
A*v        # possibly confusing answer
A.dot(v)   # [7,5] (matrix-vector mult)
```

NOTEBOOK TIME

I'll build a Python notebook demonstrating some basic features of numpy.

After lecture it will be in the [course sample code repo](#).

INDEXING AND SLICING

Numpy has powerful syntax for retrieving individual elements or collections of elements of arrays.

Most basic version: $A[i, j]$ gives the element at row i , column j for a 2D array. Similar in higher dimensions, e.g. $A[i, j, k, l]$.

Slices return *views* of part of the array, not copies.

UFUNCS

Numpy's "ufuncs" or **universal functions** are functions that can be applied directly to arrays, automatically acting on each element.

Numpy provides a *lot* of these.

Usually, ufuncs allow you to avoid explicit iteration over array elements (which is much slower).

BOOL GOTCHA

```
np.array([5,0,1])==np.array([0,0,0])
```

evaluates to

```
np.array([False, True, False])
```

and numpy arrays *do not support boolean coercion* so this cannot appear in `if`.

To test if two arrays are equal, use one of:

```
np.all(A==B)  
np.array_equal(A,B)
```

AGGREGATIONS

Numpy has operations like sum, product, max, min, all, any, that reduce array dimension.

REFERENCES

- *Python Data Science Handbook* by Jake VanderPlas
 - Bookmark it now! We'll use it for several topics.
 - [Chapter 2](#) contains the introduction to numpy.
 - There is also a print edition from O'Reilly.

REVISION HISTORY

- 2022-03-09 Last year's lecture on this topic finalized
- 2023-03-05 Updated for 2023

