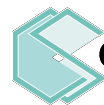


Numpy



@cambridgespark



CAMBRIDGE SPARK

Simple plan

1. Quickly introduce numpy in the console
 - a. Open up **jupyter notebook** and **tap along** with the slides
2. Practical **Jupyter Notebook** session

What is Numpy?



- A library to manipulate **arrays**
- Array \approx **indexed collection** of variables of the same type
- Types of data (**dtype**):
 - Boolean - bool
 - Integer - int32 int64
 - Float - float16 float64
 - Object - object
 - <https://docs.scipy.org/doc/numpy/user/basics.types.html>
- Type is **inferred** if not supplied: most general that 'fits'
- Optimised for computation **speed** and **ease** of use
- Other packages built on top of it e.g. **Pandas**

Practical introduction

Open up in a jupyter notebook (or ipython)...

```
$ import numpy as np
$ a1 = np.array([[1, 2, 3], [4, 5, 6]])
$ a1
> array([[1, 2, 3],
         [4, 5, 6]])
$ a1.shape
> (2, 3)
$ a1.reshape((1, 6))
> array([[1, 2, 3, 4, 5, 6]])
$ a1.reshape(6) # different?
```

Broadcasting

```
$ a2 = np.array([1, 2, 3]) # shape?
$ a3 = np.array([[10], [20], [30]]) # shape?
$ a1 + a2
> array([[2, 4, 6],
        [5, 7, 9]])
$ a1 + a3
> ... operands could not be broadcast together with shapes (2,3) (3,1)
$ a4 = a2 + a3
> array([[11, 12, 13],
        [21, 22, 23],
        [31, 32, 33]])
```

Accessing elements: indexing

```
$ a4[0, 0]
> # gets element (0, 0)

$ a4[1, :]
> # gets second *row*

$ a4[:, 2].shape
> # returns a *vector*

$ a4[1:]
> # indexes rows

$ a4[0:2]
> # doesn't include final row

$ a2[:-1]
```

Boolean masks

```
$ mask = a1 > 2
$ mask
> array([[False, False,  True],
        [ True,  True,  True]])
$ a1[mask].shape
> (4,)
$ a1[mask]
> array([3, 4, 5, 6])
```

Functions, functions, functions

Creation:

```
$ np.arange(10)
$ np.ones(100) * 10
$ np.linspace(0, 1, 100)
```

Maths:

```
$ np.sqrt(a1)
$ np.exp(a2)
$ np.log(a3)
$ np.sin(np.linspace(0, 1, 100) * 2 * np.pi)) # bonus: plot it
```


Methods, methods, methods

```
$ a1.mean()  
$ a1.sum()  
$ a1.sum(axis=0)  
$ a2.dot(a3) # a2 @ a3  
$ a1.cumsum()  
$ a2.max()  
$ a2.argmax()  
$ a1.T  
$ a1.reshape(-1)  
$ ...
```



Hands-on session

01-numpy-skeleton.ipynb

20 mins