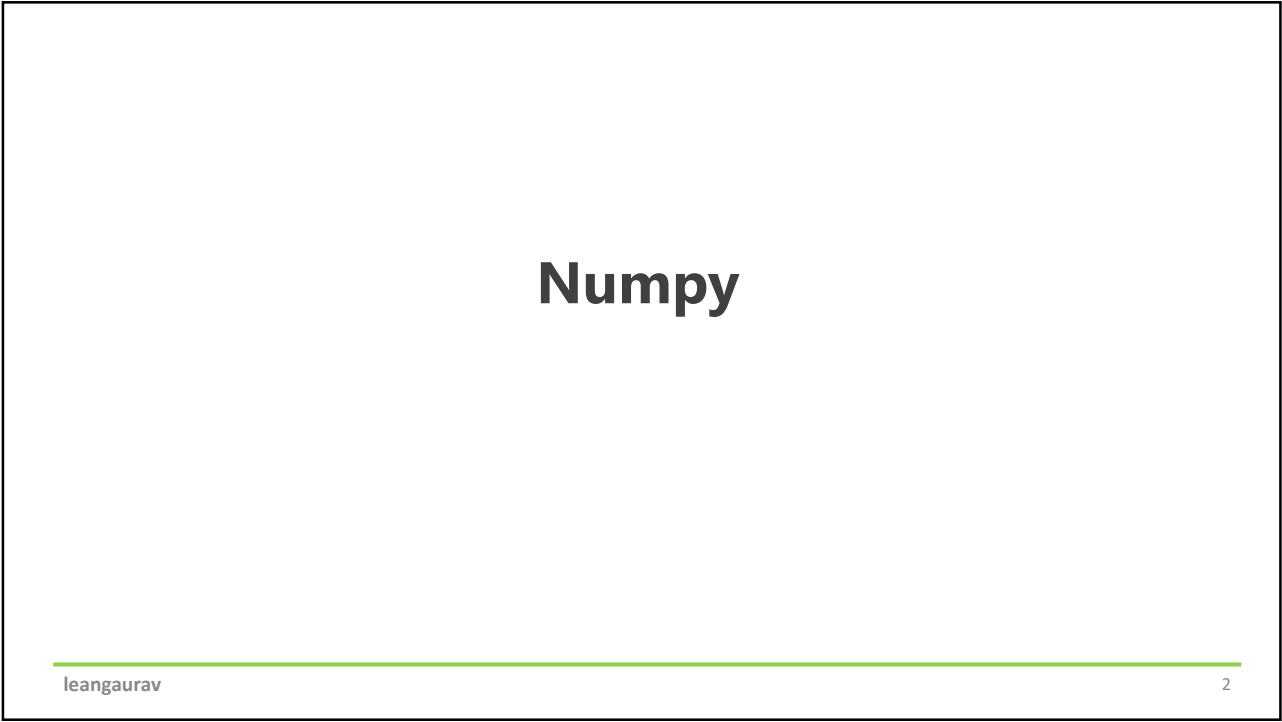




1



2

## About Numpy

**NumPy** is the fundamental package for scientific computing with Python. It contains among other things:

- 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

Source: [numpy.org](https://numpy.org)

## Data Types

### *ndarray*

- *ndarray* is similar to the array like types available in python (list, tuple)
- It internally uses C arrays to store data.

### Integer Types:

- np.int8, np.int16, np.int32, np.int64
- np.uint8, np.uint16, np.uint32, np.uint64

### Floating Types:

- np.float32, np.float64

## ndarray

### Attributes

- |            |   |
|------------|---|
| - flags    | Information about the memory layout of the array. |
| - shape    | Tuple of array dimensions.                        |
| - ndim     | Number of array dimensions.                       |
| - size     | Number of elements in the array.                  |
| - itemsize | Length of one array element in bytes.             |
| - dtype    | Data-type of the array's elements.                |
| - T        | Transposed form of array.                         |

ndarray is homogeneous. This is in contrast to the list and tuple types of Python

## Code

- `arange(start, end, step)`
- `random.randint(start, end, size = <no of elements>)` # default gives one no.
- `linspace(start, end, count)`
- `zeros(shape)` # shape single arg or tuple of shape
- `ones(shape)`

## Indexing Slicing

- Slicing works similar to normal lists
  - `array[ <row index/slice>, <column index/slice> ]`
  - `array[ 1:4, [3,4] ]`
- For multidimensional slicing use the comma syntax:
  - `array[ dim1, dim2, dim3, ..... ]`
- Boolean based indexing can be used

## Any, all and NaNs

- `any()` checks if any True value is present, it returns True then
- `all()` returns True only when all elements are True
- `isna()` returns an ndarray of same size as input, putting True/False for each element
- Nan can't be compared with other elements and each other hence all operations in numpy on NaNs return NaN

## where function

- Used to extract indexes of element where the condition is satisfied

```
>>> where( <ndarray of bools> )
```

- Ex:

```
>>> np.where ( s %2 == 0 )
```

- Result of where can be used directly in indexing other numpy arrays