Python Programming

Unit 06 – Lecture 01: NumPy Basics (ndarray, Operations, Broadcasting)

Tofik Ali

School of Computer Science, UPES Dehradun

February 14, 2026

Repository: `https://github.com/tali7c/Python-Programming`

Quick Links

Core Concepts Demo Interactive Summary

Agenda

Learning Outcomes

- Explain why NumPy is used for numerical computing

Learning Outcomes

- Explain why NumPy is used for numerical computing
- Create arrays and understand `dtype` and shape

Learning Outcomes

- Explain why NumPy is used for numerical computing
- Create arrays and understand `dtype` and shape
- Apply vectorized operations on arrays

Learning Outcomes

- Explain why NumPy is used for numerical computing
- Create arrays and understand `dtype` and shape
- Apply vectorized operations on arrays
- Explain broadcasting with examples

Why NumPy?

- Fast numerical computing (implemented in C)

Why NumPy?

- Fast numerical computing (implemented in C)
- Compact arrays with fixed data types

Why NumPy?

- Fast numerical computing (implemented in C)
- Compact arrays with fixed data types
- Vectorized operations (no Python loops for many tasks)

Why NumPy?

- Fast numerical computing (implemented in C)
- Compact arrays with fixed data types
- Vectorized operations (no Python loops for many tasks)
- Foundation for Pandas, ML, and scientific computing

List vs NumPy Array

- List: can store mixed types, slower for numeric math

```
import numpy as np
x = np.array([1, 2, 3])
print(x * 2)  # [2 4 6]
```

List vs NumPy Array

- List: can store mixed types, slower for numeric math
- NumPy array: fixed dtype, faster operations

```
import numpy as np
x = np.array([1, 2, 3])
print(x * 2)  # [2 4 6]
```

Array Creation

- `np.array([...])`

Array Creation

- `np.array([...])`
- `np.zeros((r,c))`, `np.ones((r,c))`

Array Creation

- `np.array([...])`
- `np.zeros((r,c))`, `np.ones((r,c))`
- `np.arange(start, stop, step)`

Array Creation

- `np.array([...])`
- `np.zeros((r,c))`, `np.ones((r,c))`
- `np.arange(start, stop, step)`
- `np.linspace(start, stop, num)`

Array Attributes

- shape, ndim, size, dtype

```
a = np.array([[1, 2], [3, 4]])
print(a.shape, a.ndim, a.size, a.dtype)
```

Broadcasting (Idea)

- NumPy can apply operations between arrays of different shapes

```
a = np.array([1, 2, 3])
print(a + 10)  # [11 12 13]
```

Broadcasting (Idea)

- NumPy can apply operations between arrays of different shapes
- Example: add a scalar to every element

```
a = np.array([1, 2, 3])
print(a + 10)  # [11 12 13]
```

Demo: Array Creation + Broadcasting

- File: `demo/numpy_basics_demo.py`

Demo: Array Creation + Broadcasting

- File: `demo/numpy_basics_demo.py`
- Shows:

Demo: Array Creation + Broadcasting

- File: `demo/numpy_basics_demo.py`
- Shows:
    - creating arrays

Demo: Array Creation + Broadcasting

- File: `demo/numpy_basics_demo.py`
- Shows:
  - creating arrays
  - vectorized math

Demo: Array Creation + Broadcasting

- File: `demo/numpy_basics_demo.py`
- Shows:
  - creating arrays
  - vectorized math
  - broadcasting examples

Checkpoint 1

Question: Why is `[1,2,3] * 2` different from `np.array([1,2,3]) * 2`?

Checkpoint 2

Question: What does broadcasting mean in NumPy?

Think-Pair-Share

Discuss:

- When would you still use a Python list instead of a NumPy array?

Key Takeaways

- NumPy arrays are fast and have fixed `dtype`

Key Takeaways

- NumPy arrays are fast and have fixed `dtype`
- Vectorized operations avoid explicit loops

Key Takeaways

- NumPy arrays are fast and have fixed `dtype`
- Vectorized operations avoid explicit loops
- Broadcasting applies operations across compatible shapes

Exit Question

Name any two NumPy functions used to create arrays.