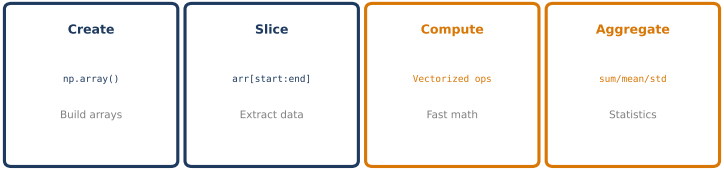
Lesson 11 Summary: NumPy Basics

Data Science with Python – Key Concepts

Data Science Program

NumPy Overview

NumPy Array Operations



Key Functions:

`dot()` | `exp()` | `log()` | `sqrt()` | `reshape()` | `percentile()`

NumPy vectorization is 10-100x faster than Python loops

NumPy is the foundation of scientific computing in Python

Creating Arrays

Ways to create NumPy arrays:

- **From list:** `np.array([1, 2, 3])`
- **Zeros/ones:** `np.zeros(5)` / `np.ones((3, 4))`
- **Range:** `np.arange(0, 10, 0.5)`
- **Linspace:** `np.linspace(0, 1, 100)`

Random arrays:

```
np.random.randn(100) # Standard normal
np.random.uniform(0, 1, size=50)
```

Arrays are homogeneous (same dtype throughout)

Vectorized Operations

Operations apply element-wise:

- **Arithmetic:** `arr * 2`, `arr1 + arr2`
- **Comparisons:** `arr > 0` (returns boolean array)
- **Functions:** `np.exp(arr)`, `np.log(arr)`

Why vectorize:

```
# Slow:  [x**2 for x in arr]
# Fast:  arr**2 (10-100x faster)
```

Avoid Python loops – use NumPy operations instead

Indexing and Slicing

Access array elements:

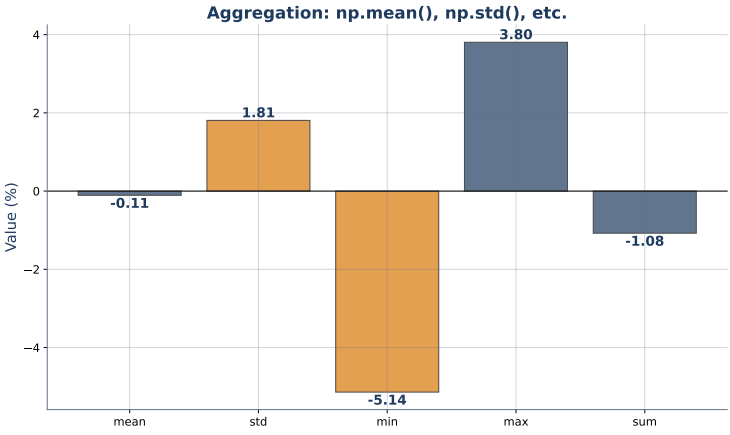
- **Single element:** `arr[0]`, `arr[2, 3]`
- **Slice:** `arr[1:5]`, `arr[::2]`
- **Boolean mask:** `arr[arr > 0]`
- **Fancy indexing:** `arr[[0, 2, 4]]`

2D slicing:

```
matrix[0, :]  # First row
matrix[:, 1] # Second column
```

NumPy slicing returns views (not copies) by default

Aggregation Functions



Aggregate along axis=0 (columns) or axis=1 (rows)

Mathematical Functions

Common NumPy functions:

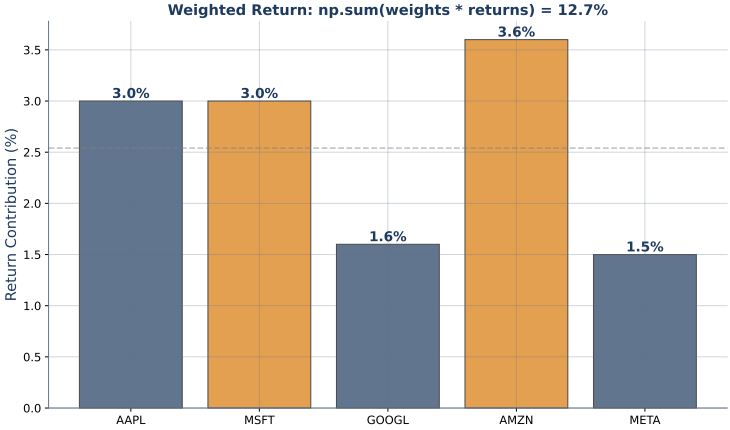
- **Exponential:** `np.exp()`, `np.log()`, `np.log10()`
- **Power:** `np.sqrt()`, `np.power(arr, 2)`
- **Statistics:** `np.percentile()`, `np.corrcoef()`
- **Clipping:** `np.clip(arr, min, max)`

Finance example:

```
log_returns = np.log(prices[1:]  / prices[:-1])
```

Universal functions (ufuncs) are optimized C implementations

Portfolio Calculations



np.dot() computes weighted portfolio returns efficiently

Broadcasting

NumPy automatically aligns array dimensions:

- **Scalar + array:** `arr + 5`
- **Row + matrix:** Each row gets the vector added
- **Rules:** Dimensions must match or be 1

Example:

```
prices = np.array([[100, 150], [102, 148]])
fees = np.array([0.01, 0.02]) # Per asset
net = prices * (1 - fees) # Broadcasting
```

Broadcasting enables concise array operations

Quick Reference

Essential NumPy Operations:

Operation	Syntax
Create array	`np.array([1, 2, 3])`
Zeros/ones	`np.zeros(n)` / `np.ones(n)`
Statistics	`np.mean()`, `np.std()`, `np.sum()`
Math functions	`np.exp()`, `np.log()`, `np.sqrt()`
Dot product	`np.dot(a, b)` or `a @ b`
Reshape	`arr.reshape((m, n))`
Boolean filter	`arr[arr > 0]`
Correlation	`np.corrcoef(a, b)`

NumPy is essential for efficient numerical computing