NumPy

An overview of NumPy, the array that nearly every scientific Python library builds on, with vectorized math, broadcasting, and linear algebra.

blog.thecoatlessprofessor.com
updated 05/29/2026

Create Arrays

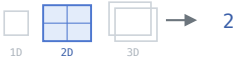
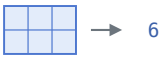
Turn lists, ranges, and constants into ndarrays.

PURPOSE	COMMAND	RESULT
From a Python list.	`np.array([1, 2, 3])`	
All zeros, given shape.	`np.zeros((2, 3))`	
All ones, given shape.	`np.ones(3)`	
Filled with a constant.	`np.full((2, 2), 7)`	
Evenly spaced by step.	`np.arange(0, 10, 2)`	
Evenly spaced by count.	`np.linspace(0, 1, 5)`	

an ndarray is a fixed-size, same-type block laid out contiguously in memory

Dtypes & Attributes

Every array has a shape and one element type.

PURPOSE	COMMAND	RESULT
Shape (rows, cols).	`a.shape`	
Number of dimensions.	`a.ndim`	
Total element count.	`a.size`	
The element type.	`a.dtype`	
Set type on creation.	`np.array([1, 2, 3], dtype=np.float64)`	
Convert (copy) to a type.	`a.astype(np.float32)`	

dtype controls memory and precision; astype returns a converted copy

Index & Slice

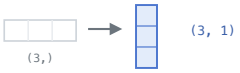
Select elements, rows, columns, and masks; views, not copies.

PURPOSE	COMMAND	RESULT
Single element [row, col].	`m[1, 2]`	
A whole row.	`m[0]`	
A whole column.	`m[:, 1]`	
Slice a range / every other.	`a[2:5] a[::2]`	
Boolean mask (filter).	`a[a > 5]`	
Assign into a selection.	`a[a > 5] = 0`	

basic slices return a view (writes propagate); masks return a copy

Reshape & Combine

Rearrange the same data; join arrays along an axis.

PURPOSE	COMMAND	RESULT
Reshape (infer one dim).	`a.reshape(2, 3)`	
Flatten to 1D.	`a.ravel()`	
Transpose (swap axes).	`a.T`	
Add a length-1 axis.	`a[:, np.newaxis]`	
Stack rows / columns.	`np.vstack([a,b]) np.hstack([a,b])`	
Join along an axis.	`np.concatenate([a, b], axis=0)`	

reshape, ravel, T, newaxis move zero data; concatenate grows the array

Element-wise Math

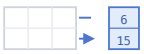
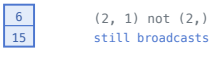
Operators and ufuncs apply cell-by-cell, vectorized.

PURPOSE	COMMAND	RESULT
Add / multiply arrays.	`a + b a * b`	
Scalar broadcast.	`a * 2`	
Square root / exp.	`np.sqrt(a) np.exp(a)`	
Round / clip values.	`np.clip(a, 2, 8)`	
Conditional select.	`np.where(a > 3, a, 0)`	
Matrix / dot product.	`a @ bnp.dot(a, b)`	

ufuncs run in compiled C; @ and np.dot contract a dimension instead

Reduce Over Axes

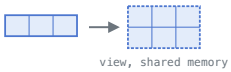
Collapse an array to summaries; choose the axis.

PURPOSE	COMMAND	RESULT
Sum / mean of all.	`a.sum() a.mean()`	
Reduce down columns.	`a.sum(axis=0)`	
Reduce across rows.	`a.sum(axis=1)`	
Spread (std / min / max).	`a.std() .min() .max()`	
Position of extreme.	`a.argmax()`	
Keep dims for broadcasting.	`a.sum(axis=1, keepdims=True)`	

axis=0 reduces down rows; axis=1 across columns; none -> a scalar

Broadcasting

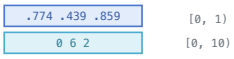
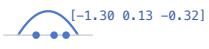
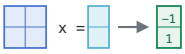
NumPy stretches length-1 dimensions so shapes line up.

PURPOSE	COMMAND	RESULT
Scalar across an array.	`a + 10`	
Row vector down a matrix.	`a + row`	
Column vector across.	`a + col`	
Shape alignment rule.	`(3,) vs (2, 3)`	
Outer product via axes.	`a[:, None] * b[None, :]`	
Explicit stretch (no copy).	`np.broadcast_to(a, (2, 3))`	

NumPy compares shapes from the right and stretches length-1 dims to match

Random & Linear Algebra

Draw reproducible randoms; solve and decompose matrices.

PURPOSE	COMMAND	RESULT
Seeded generator.	`rng =np.random.default_rng(42)`	
Uniform / integer draws.	`rng.random(3) rng.integers(0,10,3)`	
Normal draws.	`rng.normal(0, 1, 3)`	
Solve a linear system.	`np.linalg.solve(A, b)`	
Inverse / determinant.	`np.linalg.inv(A) .det(A)`	
Save / load .npy.	`np.save("a.npy", A) np.load(...)`	

default_rng(seed) makes draws reproducible; np.linalg covers everyday matrices