NumPy 快速参考

数组创建、数学运算、线性代数等

数组创建

从列表创建

```
import numpy as np
a = np.array([1, 2, 3])        # 1D
b = np.array([[1, 2], [3, 4]])  # 2D
```

内置构造函数

```
np.zeros((2, 3))     # 2x3 of zeros
np.ones((3, 3))      # 3x3 of ones
np.eye(4)            # 4x4 identity matrix
np.arange(0, 10, 2)  # [0, 2, 4, 6, 8]
np.linspace(0, 1, 5) # 5 evenly spaced
```

数组属性

a.shape	维度元组：**(3, 4)**
a.ndim	维度数量
a.size	元素总数
a.dtype	数据类型：**float64**、**int32** 等

索引与切片

基本索引

```
a = np.array([[1, 2, 3], [4, 5, 6]])
a[0, 1]    # 2 (row 0, col 1)
a[1]       # [4, 5, 6] (row 1)
a[:, 0]    # [1, 4] (all rows, col 0)
```

切片

```
a[0, 1:]   # [2, 3] (row 0, col 1 onward)
a[:, :2]   # first 2 columns
a[::2]     # every other row
```

布尔索引

```
a = np.array([10, 20, 30, 40])
a[a > 15]   # [20, 30, 40]
a[a % 20 == 0]  # [20, 40]
```

数组运算

元素级运算

```
a = np.array([1, 2, 3])
a + 10    # [11, 12, 13]
a * 2     # [2, 4, 6]
a ** 2    # [1, 4, 9]
a + a     # [2, 4, 6]
```

比较

```
a = np.array([1, 2, 3, 4])
a > 2         # [False, False, True, True]
np.where(a > 2, a, 0)  # [0, 0, 3, 4]
```

聚合

a.sum()	所有元素之和
a.mean()	算术平均值
a.std()	标准差
a.min() / a.max()	最小值 / 最大值
a.argmin() / a.argmax()	最小值 / 最大值的索引
a.cumsum()	累计求和

添加 `axis=0`（按列）或 `axis=1`（按行）可按轴计算

数学函数

常用函数

np.sqrt(a)	每个元素的平方根
np.abs(a)	绝对值
np.exp(a)	每个元素的 e^x
np.log(a)	自然对数（ln）
np.log10(a)	以 10 为底的对数
np.sin(a) / np.cos(a)	三角函数（弧度）
np.round(a, 2)	保留 2 位小数
np.clip(a, lo, hi)	将值限制在 [lo, hi] 范围

线性代数

矩阵运算

```
A = np.array([[1, 2], [3, 4]])
B = np.array([[5, 6], [7, 8]])
A @ B             # matrix multiply
np.dot(A, B)      # same as A @ B
A.T               # transpose
```

分解与求解

```
np.linalg.inv(A)       # inverse
np.linalg.det(A)       # determinant
np.linalg.eig(A)       # eigenvalues/vectors
np.linalg.solve(A, b)  # solve Ax = b
```

随机数

随机数生成

```
rng = np.random.default_rng(42)  # seeded
rng.random((2, 3))     # uniform [0, 1)
rng.integers(1, 10, 5) # 5 ints in [1, 10)
rng.normal(0, 1, 100)  # 100 from N(0,1)
rng.choice([1, 2, 3], size=2)  # sample
```

旧版 API

```
np.random.seed(42)
np.random.rand(3, 3)     # uniform 3x3
np.random.randn(3, 3)    # standard normal
np.random.shuffle(arr)   # in-place shuffle
```

形状变换

形状操作

```
a = np.arange(12)
a.reshape(3, 4)    # 3x4 matrix
a.reshape(3, -1)   # infer columns
a.flatten()        # back to 1D (copy)
a.ravel()          # back to 1D (view)
```

堆叠与分割

```
np.vstack([a, b])  # stack vertically
np.hstack([a, b])  # stack horizontally
np.concatenate([a, b], axis=0)
np.split(a, 3)     # split into 3 parts
```

广播

广播原理

```
a = np.array([[1, 2, 3],
              [4, 5, 6]])  # shape (2,3)
b = np.array([10, 20, 30]) # shape (3,)
a + b  # b broadcasts to (2,3)
```

广播规则

规则 1	向较短的 shape 前面补 1，直到维度数相同
规则 2	维度匹配条件：相等或其中一个为 1
规则 3	大小为 1 的维度扩展以匹配另一方

文件 I/O

NumPy 二进制格式

```
np.save("data.npy", arr)     # single array
arr = np.load("data.npy")
np.savez("data.npz", a=x, b=y) # multiple
d = np.load("data.npz"); d["a"]
```

文本文件

```
np.savetxt("data.csv", arr, delimiter=",")
arr = np.loadtxt("data.csv", delimiter=",")
arr = np.genfromtxt("data.csv", delimiter=",",
                    skip_header=1)
```

常见模式

归一化到 [0, 1]

```
normalized = (a - a.min()) / (a.max() - a.min())
```

欧式距离

```
dist = np.sqrt(np.sum((a - b) ** 2))
# or: np.linalg.norm(a - b)
```

唯一值与计数

```
vals, counts = np.unique(a, return_counts=True)
dict(zip(vals, counts))
```

排序

```
np.sort(a)            # sorted copy
idx = np.argsort(a)   # indices that sort
a[idx]                # apply sort order
```