# Referencia Rápida de NumPy

Creación de arrays, matemáticas, álgebra lineal y más

## Creación de Arrays

### Desde Listas

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

### Constructores Integrados

```
np.zeros((2, 3))     # 2x3 de ceros
np.ones((3, 3))      # 3x3 de unos
np.eye(4)            # identidad 4x4
np.arange(0, 10, 2)  # [0, 2, 4, 6, 8]
np.linspace(0, 1, 5) # 5 valores equiespaciados
```

### Propiedades del Array

| | |
|---|---|
| **`a.shape`** | Dimensiones como tupla: **`(3, 4)`** |
| **`a.ndim`** | Número de dimensiones |
| **`a.size`** | Total de elementos |
| **`a.dtype`** | Tipo de dato: **`float64`**, **`int32`**, etc. |

## Indexación y Cortes

### Indexación Básica

```
a = np.array([[1, 2, 3], [4, 5, 6]])
a[0, 1]    # 2 (fila 0, col 1)
a[1]       # [4, 5, 6] (fila 1)
a[:, 0]    # [1, 4] (todas las filas, col 0)
```

### Cortes

```
a[0, 1:]   # [2, 3] (fila 0, col 1 en adelante)
a[:, :2]   # primeras 2 columnas
a[::2]     # cada dos filas
```

### Indexación Booleana

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

## Operaciones con Arrays

### Operaciones Elemento a Elemento

```
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]
```

### Comparación

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

### Agregación

| | |
|---|---|
| **`a.sum()`** | Suma de todos los elementos |
| **`a.mean()`** | Media aritmética |
| **`a.std()`** | Desviación estándar |
| **`a.min()` / `a.max()`** | Valor mínimo / máximo |
| **`a.argmin()` / `a.argmax()`** | Índice del mínimo / máximo |
| **`a.cumsum()`** | Suma acumulada |

Agrega `axis=0` (columnas) o `axis=1` (filas) para resultados por eje

## Funciones Matemáticas

### Funciones Comunes

| | |
|---|---|
| **`np.sqrt(a)`** | Raíz cuadrada de cada elemento |
| **`np.abs(a)`** | Valor absoluto |
| **`np.exp(a)`** | e^x para cada elemento |
| **`np.log(a)`** | Logaritmo natural (ln) |
| **`np.log10(a)`** | Logaritmo en base 10 |
| **`np.sin(a)` / `np.cos(a)`** | Funciones trigonométricas (radianes) |
| **`np.round(a, 2)`** | Redondear a 2 decimales |
| **`np.clip(a, lo, hi)`** | Limitar valores al rango [lo, hi] |

## Álgebra Lineal

### Operaciones Matriciales

```
A = np.array([[1, 2], [3, 4]])
B = np.array([[5, 6], [7, 8]])
A @ B              # multiplicación matricial
np.dot(A, B)       # igual que A @ B
A.T                # transpuesta
```

### Descomposición y Resolución

```
np.linalg.inv(A)       # inversa
np.linalg.det(A)       # determinante
np.linalg.eig(A)       # valores/vectores propios
np.linalg.solve(A, b)  # resolver Ax = b
```

## Aleatoriedad

### Generación de Números Aleatorios

```
rng = np.random.default_rng(42)  # semillado
rng.random((2, 3))     # uniforme [0, 1)
rng.integers(1, 10, 5) # 5 enteros en [1, 10)
rng.normal(0, 1, 100)  # 100 de N(0,1)
rng.choice([1, 2, 3], size=2)  # muestra
```

### API Heredada

```
np.random.seed(42)
np.random.rand(3, 3)     # uniforme 3x3
np.random.randn(3, 3)    # normal estándar
np.random.shuffle(arr)   # mezcla en sitio
```

## Reorganización

### Manipulación de Forma

```
a = np.arange(12)
a.reshape(3, 4)    # matriz 3x4
a.reshape(3, -1)   # inferir columnas
a.flatten()        # volver a 1D (copia)
a.ravel()          # volver a 1D (vista)
```

### Apilado y División

```
np.vstack([a, b])  # apilar verticalmente
np.hstack([a, b])  # apilar horizontalmente
np.concatenate([a, b], axis=0)
np.split(a, 3)     # dividir en 3 partes
```

## Difusión (Broadcasting)

### Cómo Funciona el Broadcasting

```
a = np.array([[1, 2, 3],
              [4, 5, 6]])  # forma (2,3)
b = np.array([10, 20, 30]) # forma (3,)
a + b  # b se difunde a (2,3)
```

### Reglas

| | |
|---|---|
| **`Regla 1`** | Se anteponen 1s a la forma más corta hasta que los rangos coincidan |
| **`Regla 2`** | Las dimensiones coinciden si son iguales o una es 1 |
| **`Regla 3`** | Las dimensiones de tamaño 1 se estiran para coincidir |

## E/S de Archivos

### Binario NumPy

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

### Archivos de Texto

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

## Patrones Comunes

### Normalizar a [0, 1]

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

### Distancia Euclidiana

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

### Valores Únicos y Conteos

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

### Ordenamiento

```
np.sort(a)            # copia ordenada
idx = np.argsort(a)   # índices que ordenan
a[idx]                # aplicar orden
```