🔑 NumPy Essential Cheatsheet

For Mathematical operations *Focus on the 90% most-used concepts*

🔥 NumPy Essentials (Numerical Computing Foundation)

1. Array Basics

```
 np.array(list/tuple) # Create an array
```

```
.dtype # Data type (e.g., int64, float32, bool, object)
```

```
.shape # Dimensions (e.g., (3,4) = 3 rows, 4 columns)
```

```
.size # Total number of elements
```

```
.ndim # Number of dimensions (1D, 2D, 3D)
```

Creation Functions:

```
np.zeros((m,n)) # Array filled with 0s
```

```
np.ones((m,n)) # Array filled with 1s
```

```
np.arange(start, stop, step) # Range-based array
```

```
np.linspace(start, stop, num=50) # Evenly spaced values
```

```
np.eye(n) # Identity matrix
```

```
np.full(shape, value) # Fill with specific value
```

```
np.empty(shape) # Fast uninitialized array
```

---

2. Indexing & Slicing

- **Basic Indexing:**

```
  a[0] # First element
```

```
  a[1:5] # Slice elements
```

```
  a[0,1] # 2D indexing
```

```
  a[:, 2] # Select column
```

- **Boolean Indexing:**

```
  a[a > 5]
  a[a % 2 == 0]
```

- **Fancy Indexing:**

```
  a[[0,2,4]]
  a[[0,1], [2,3]]
```

- **Advanced:**

```
np.newaxis or `None` # Add new dimension
```

---

3. Element-wise Operations & Broadcasting

- Arithmetic operations: `+`, `-`, `*`, `/`, `**`, `//`, `%`
- **Broadcasting:** Operations between arrays of different shapes Example: `(3,1) + (1,4) # (3,4)`
- **Aggregate Functions:**

```
np.sum(a, axis=0/1)
```

```
np.mean(a)
```

```
np.median(a)
```

```
np.min(a), np.max(a)
```

```
np.argmin(a), np.argmax(a)
```

```
np.std(a), np.var(a)
```

```
np.percentile(a, q=95)
```

---

4. Linear Algebra (Very Important for ML)

```
np.dot(a,b) or a @ b # Matrix multiplication
```

```
np.matmul(a,b)
```

```
np.linalg.inv(A) # Matrix inverse
```

```
np.linalg.det(A) # Determinant
```

```
np.linalg.eig(A) # Eigenvalues & eigenvectors
```

```
np.linalg.norm(x, ord=2) # Vector/matrix norm
```

```
np.linalg.svd(A) # Singular Value Decomposition
```

```
np.linalg.solve(A, b) # Solve linear equations
```

---

5. Random Number Generation

```
np.random.seed(42) # Reproducibility
```

```
np.random.rand(shape) # Uniform distribution [0,1)
```

```
np.random.randn(shape) # Standard normal distribution
```

```
np.random.randint(low, high, size)
```

```
np.random.choice(a, size, p=probabilities)
```

```
np.random.permutation()
```

```
np.random.shuffle()
```

Modern Approach:

```
rng = np.random.default_rng()` # Recommended method
```

---

6. Reshaping & Manipulation

```
a.reshape(new_shape) # Reshape array (view, fast)
```

```
a.ravel() # Flatten to 1D (view)
```

```
a.flatten() # Flatten to 1D (copy)
```

```
np.transpose(a) or a.T # Transpose
```

Joining Arrays:

```
np.concatenate((a,b), axis=0/1)
```

```
np.vstack((a,b))
```

```
np.hstack((a,b))
```

```
np.column_stack((a,b))
```

Splitting Arrays:

```
np.split(a, indices_or_sections)`
```

Repeating Arrays:

```
np.tile(a, reps)
np.repeat(a, repeats)
```
