Build Your Own MiniNumPy Library

Objective

The goal of this project is to design and implement a simplified version of **NumPy**, called `MiniNumPy`, to understand how numerical computing and linear algebra libraries are built from scratch. Students will implement core data structures, array operations, and linear algebra routines.

By the end, students should:

- Understand how arrays are represented in memory.
- Implement basic array manipulation (reshape, transpose, slicing).
- Write elementwise and matrix operations.
- Explore algorithms for linear algebra (determinant, inverse, eigenvalues).

Project Tasks

Part 1: Core Array Class

- Implement a class `Array` that wraps a Python list (or nested lists).
- Store attributes: `.data`, `.shape`, `.ndim`, `.size`.
- Add methods:
  - `reshape(new_shape)`
  - `transpose()`
  - `__str__` for pretty printing

Part 2: Array Creation

- Implement helper functions:
  - `array(list_or_nested_list)`
  - `zeros(shape)`
  - `ones(shape)`
  - `eye(n)`
  - `arange(start, stop, step)`
  - `linspace(start, stop, num)`

Part 3: Elementwise Operations

- Overload Python operators (`+`, `-`, `*`, `/`, `**`).

- Implement elementwise functions:
    - `exp`, `log`, `sqrt`, `abs`.
- Implement reductions:
    - `sum`, `mean`, `min`, `max`, `argmin`, `argmax`.

Part 4: Linear Algebra Module (`minilinalg`)

- Implement matrix/vector operations:
    - `dot(a, b)` – dot product / matrix multiply.
    - `matmul(a, b)` – general matrix multiplication (@ operator).
    - `norm(a)` – vector/matrix norm.
- Implement basic factorizations/solvers:
    - `det(a)` – determinant (via recursion or LU).
    - `inv(a)` – matrix inverse.
    - `eig(a)` – eigenvalues and eigenvectors (bonus).

Part 5: Applications (Mini-Projects)

Students must demonstrate their library with **practical applications**:

1. Image manipulation (grayscale filter or rotation using matrices).
2. 2D transformation: scale, rotate, and shear a set of points.

Deliverables

1. Source code of `MiniNumPy` (`mininumpy/array.py`, `mininumpy/linalg.py`).
2. A **report (5–10 pages)** explaining design choices and algorithms.
3. A **demo notebook** showing use cases and comparisons with real NumPy.

row x cols

~~cols x rows~~