Introduction to Programming for AI

Wigner Summer Camp

Data and Compute Intensive Sciences Research Group

Balázs, Paszkál, Vince, Levente, Antal
Éva, Hajni

7-11 July 2025

Course Overview

This course will cover essential programming tools for AI:

- Core Python programming
- Numerical computing with NumPy
- Data manipulation with pandas
- Data visualization with matplotlib
- Deep learning basics with PyTorch

Each topic has:

- Lecture notebook (e.g., python_basics.ipynb)
- Solution notebook (e.g., python_basics_solutions.ipynb)

Why Python for AI?

- **Readability:** Simple syntax similar to pseudocode.
- **Ecosystem:** Rich collection of libraries for scientific computing.
- **Community:** Large user base and extensive documentation.
- **Flexibility:** Supports both prototyping and production.

Example (Simple Python)

```
# Calculate factorial
def factorial(n):
return 1 if n == 0 else n * factorial(n-1)
```

Python Basics

Core concepts you'll learn about:

- Variables and data types (int, float, str, bool).
- Control flow (if-else, for/while loops).
- Functions and lambda expressions.
- Lists, tuples, dictionaries, sets.
- Object-oriented programming basics.

Example (List Comprehension)

```python
squares = [x**2 for x in range(10)]
```

NumPy: Numerical Computing

- **What:** Fundamental package for scientific computing.
- **Why:** Efficient array operations and linear algebra.
- **Key features:**
  - N-dimensional array object.
  - Broadcasting functions.
  - Linear algebra, Fourier transform, random number capabilities.

Example (NumPy array)

```python
import numpy as np
a = np.array([[1, 2], [3, 4]])
b = np.ones((2,2))
c = a + b  # Element-wise addition
```

pandas: Data Analysis

- **What:** Powerful data manipulation library.
- **Why:** Essential for data cleaning and preprocessing.
- **Key features:**
  - DataFrame object (like Excel tables).
  - Handling missing data.
  - Merging and joining datasets.
  - Time series functionality.

Example (pandas DataFrame)

```python
import pandas as pd
data = {'Name': ['Alice', 'Bob'], 'Age': [25,
    30]}
df = pd.DataFrame(data)
```

matplotlib: Data Visualization

- **What:** Comprehensive 2D plotting library.
- **Why:** Visualize data and model results.
- **Key features:**
  - Publication-quality figures.
  - Various plot types (line, bar, scatter, histogram).
  - Customizable styling.

Example (Simple plot)

```
import matplotlib.pyplot as plt
plt.plot([1, 2, 3], [1, 4, 9])
plt.xlabel('X␣axis')
plt.ylabel('Y␣axis')
```

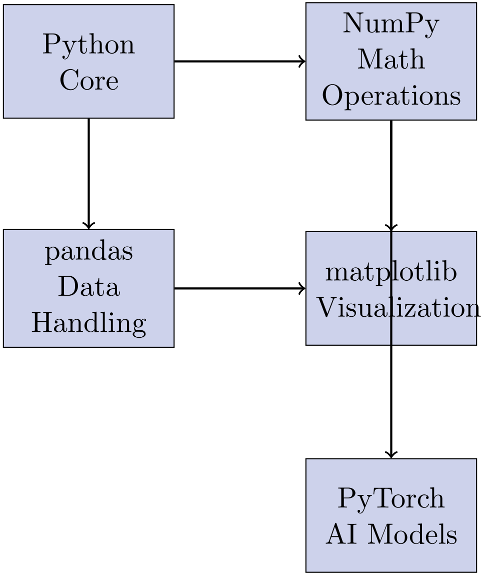
PyTorch: Deep Learning

- **What:** Open-source machine learning library.
- **Why:** Flexible architecture for research and production.
- **Key features:**
  - Tensor computation with GPU acceleration.
  - Automatic differentiation.
  - Neural network building blocks.

Example (Simple tensor operation)

```python
import torch
x = torch.tensor([[1, 2], [3, 4.]])
y = torch.matmul(x, x)  # Matrix multiplication
```

How These Tools Work Together



Practical Workflow Example

1. Load and clean data with pandas.
2. Perform numerical operations with NumPy.
3. Visualize results with matplotlib.
4. Build and train models with PyTorch.
5. Analyze results and iterate.

Example (AI workflow)

```
# 1. Load data
data = pd.read_csv('dataset.csv')

# 2. Preprocess
X = data[['feature1', 'feature2']].values
y = data['target'].values

# 3. Build model
model = torch.nn.Linear(2, 1)
```