The Mathematics of Complex Systems: Theory and Applications
Day 1: Introduction to Python

Danillo Barros de Souza

Basque Center for Applied Mathematics

Course Roadmap

- **Day 1:** Introduction to Python and scientific computing
- **Day 2:** Numerical solution of ODEs
  - Euler's method
  - Stability and accuracy
- **Day 3:** Higher-order numerical methods
  - 4th-order Runge–Kutta method
  - Error comparison with Euler's method
- **Day 4:** Parameter estimation
  - Least squares fitting
  - Applications to epidemic models

Today's Class

- Introduction to Python
- Installing Python and package managers
- Core Python syntax
- Numerical computing with NumPy
- Data analysis with Pandas

What is Python?

- High-level, interpreted programming language
- Emphasis on readability and simplicity
- Widely used in scientific computing and data analysis

Installing Python

- Conda (recommended for scientific computing)
- pip (standard Python package manager)

For Loops

```
for i in range(5):
    print(i)
```

```
values = [1, 2, 3, 4]
for v in values:
    print(v)
```

The range Function

- Generates a sequence of integers
- Commonly used in loops
- Memory-efficient (does not store the full list)

```
range(5)              # 0, 1, 2, 3, 4
range(1, 6)           # 1, 2, 3, 4, 5
range(0, 10, 2)       # 0, 2, 4, 6, 8
```

The np.arange Function

- NumPy equivalent of `range` for arrays
- Supports floating-point steps
- Returns a NumPy array

```
import numpy as np

np.arange(5)
np.arange(0, 1, 0.2)
np.arange(1, 10, 2)
```

Lambda Functions

- Anonymous (one-line) functions
- Useful for short mathematical expressions
- Common in numerical and data analysis

```
f = lambda x: x**2
f(3)

g = lambda x, y: x + y
g(2, 5)
```

NumPy: Numerical Computing

- Efficient arrays and vectorized operations
- Fundamental for scientific computing

```
import numpy as np

a = np.array([1, 2, 3])
b = np.array([4, 5, 6])

a + b
a * b
np.dot(a, b)
```

Summary

- Python basics and control structures
- `range` vs `np.arange`
- Lambda functions
- NumPy and Pandas fundamentals

Defining a Python Function

- Functions are defined using the `def` keyword
- They allow code reuse and clearer structure
- Can take multiple inputs and return values

```
def square(x):
    return x**2

result = square(5)
print(result)
```

Questions?