MAT 835: Scientific Computing with Python

John Karuitha

1 Purpose of the Course

The purpose of this course is to introduce learners to the fundamentals of computing. This outline focuses on scientific computing in Python, deemed the most relevant and effective tool for the field.

2 Expected Learning Outcomes

By the end of the course, the learner should be able to:

1. Document, debug, and test developed code.
2. Optimize Python code.
3. Demonstrate acquisition of fundamental knowledge in key aspects of scientific computing.
4. Mathematically map formulated algorithms to Python code.
5. Write small Python programs using variables, mathematical, and logical operators.
6. Demonstrate critical thinking, integrity, and ethical decision-making through logical reasoning, fostering intellectual honesty, responsibility, and professionalism.

---

3 Course Outline

3.1 Fundamentals of Computing

- How computers work: Hardware and software essentials
- Program code: Compilation vs. interpretation
- Setting up and running Python programs

3.2 Programming Basics and Introduction to Scientific Computing with Python

- Writing and running Python code
- Input/output operations
- Overview of scientific computing and its applications
- Why Python for scientific computing?
- Setting up the Python development environment: Anaconda, Jupyter Notebooks, and IDEs
- Introduction to Python libraries for scientific computing (NumPy, SciPy, Pandas, Matplotlib, etc.)

3.3 Variables and Data Types

- Variable declaration and assignment
- Built-in data types: int, float, str, list, tuple, dict, set
- Type conversions and type checking

3.4 Mathematical and Logical Operators

- Arithmetic operations: Addition, subtraction, multiplication, division, modulus, and exponentiation
- Relational operators: Greater than, less than, equality checks
- Logical operators: and, or, not
- Operator precedence and associativity

3.5 Conditionals and Loops

- Conditional statements: if, else, elif
- Iterative structures: for loops, while loops
- Nested conditionals and loops

3.6 Procedural Programming

- Writing modular programs
- Function definition and usage
- Parameter passing and return values
- Scope and lifetime of variables

3.7 Python Modules and Their Use

- Installing Python modules: pip and conda
- Overview and use of essential libraries:
  - NumPy for numerical computations
  - Pandas for data manipulation
  - Matplotlib and Seaborn for data visualization
  - Scikit-learn for machine learning basics

3.8 Object-Oriented Programming in Python

- Introduction to object-oriented programming (OOP) concepts
- Classes and objects in Python
- Attributes and methods
- Encapsulation, inheritance, and polymorphism
- Using OOP in scientific computing
- Case studies and real-world applications

3.9 Python for Scientific Computing

- Numerical computations with NumPy
- Statistical analysis and data manipulation with Pandas
- Solving equations and optimization using SciPy

3.10 Data Analysis with Python

- Data wrangling and cleaning with Pandas
- Exploratory data analysis (EDA)
- Descriptive statistics and data aggregation
- Handling missing data
- Working with time series data

3.11 Data Visualization

- Creating plots with Matplotlib
- Customizing plots (labels, legends, grids, styles)
- Introduction to Seaborn for statistical data visualization

3.12 Exceptions, Debugging and Testing

- Types of errors: Syntax, runtime, and logical errors
- Raising and Handling Exceptions in Python
- Debugging tools and techniques
- Writing and using test cases
- Understanding Python error messages and stack traces

3.13 Ethics and Professionalism in Scientific Computing

- Ethical considerations in software development
- Data privacy and security
- Responsible usage of computational tools

---

4 Instructional Methodologies

- **Lectures:** Foundational knowledge and theoretical explanations
- **Research and Seminar Presentations:** Students present assigned topics to peers
- **Guided Practical Work:** Hands-on sessions to practice concepts
- **E-learning Platform Discussions:** Asynchronous and synchronous discussions
- **Discovery Learning:** Problem-solving through exploration
- **Group-Based Learning:** Collaborative projects and exercises
- **Library Search:** In-depth exploration of additional resources

---

5 Instructional Materials and Equipment

- Laptop/Tablet
- LCD Projectors
- Resource Persons
- Smartphone
- Library and E-resources
- Computers with MATLAB and Python installed
- Reference Texts and Scientific Publications

---

6 Learner Assessment

Type	Weighting (%)
Continuous Assessment Tests	40
- Term Paper	10
- Project	10
- Seminar Paper Presentations	10
- Individual Work	10
End of Semester Examination	60
Total	100

---

7 Suggested Reading Texts and Resources

1. **"Python for Data Analysis" by Wes McKinney**
2. **"Python Data Science Handbook" by Jake VanderPlas**
3. **"Introduction to Scientific Computing with Python" by Hans Petter Langtangen**
4. Online resources: Python documentation, NumPy and Pandas official guides
5. Journals and papers relevant to scientific computing and Python applications