هيئة تقويم التعليم والتدريب
Education & Training Evaluation Commission
اعتماد
NCAAA
2024
TP-153

Course Specification

(Bachelor)

Course Title:	Scientific Computing
Course Code:	SE4714
Program:	BSc in Software Engineering
Department:	Software Engineering
College:	College of Computing
Institution:	Umm Al Qura University
Version:	1.0
Last Revision Date:	22/04/2025

Table of Contents

A. General information about the course:

1. Course Identification

1. Credit hours: (3)					
2. Course type					
A.	☐University	☐College	☒ Department	☐Track	☐Others
B.	☐ Required		☒ Elective		
3. Level/year at which this course is offered: (3rd year/ 5th or 6th level) or (4th year/8th level)					
4. Course General Description:					
This course introduces students to the principles and practices of scientific computing, focusing on solving mathematical, engineering, and real-world problems using computational methods. It bridges the gap between theoretical mathematics and practical implementation, equipping students with the tools and techniques necessary for numerical analysis, data modeling, and simulation. Through hands-on programming exercises and projects, students will gain proficiency in Python, MATLAB, and other scientific computing tools while understanding the mathematical concepts underpinning these methods. Topics include numerical error analysis, linear algebra applications, optimization, differential equations, and high-performance computing.					
5. Pre-requirements for this course (if any)**:**					
CS2105 - Data Structures and Algorithms with Java					
6. Co-requisites for this course (if any)**:**					
N/A					
7. Course Main Objective(s):					
By the end of this course, students will be able to: 1. Understand the fundamental concepts of scientific computing, including numerical methods and their limitations. 2. Apply numerical techniques to solve linear and nonlinear systems of equations, optimize functions, and perform data fitting. 3. Use Python and related libraries (e.g., NumPy, SciPy, Matplotlib) to implement and analyze scientific computing algorithms. 4. Solve ordinary and partial differential equations using numerical methods.					

5. Develop, test, and debug computational models for real-world engineering and scientific problems.
6. Perform scientific visualization and communicate results effectively through graphical representation.
7. Explore advanced topics such as parallel computing, Fourier analysis, and Monte Carlo simulations.

2. Teaching mode (mark all that apply)

No	Mode of Instruction	Contact Hours	Percentage
1	Traditional classroom	45	100%
2	E-learning	0	0
3	Hybrid • Traditional classroom • E-learning	0	0
4	Distance learning	0	0

3. Contact Hours (based on the academic semester)

No	Activity	Contact Hours
1.	**Lectures**	45
2.	**Laboratory/Studio**	0
3.	**Field**	0
4.	**Tutorial**	0
5.	**Others (specify)**	0
Total		45

B. Course Learning Outcomes (CLOs), Teaching Strategies and Assessment Methods

Code	Course Learning Outcomes	Code of PLOs aligned with the program	Teaching Strategies	Assessment Methods
1.0	**Knowledge and understanding**			
1.1	Grasp the core principles of scientific computing, including its role in solving mathematical and engineering problems.	K1	Lecture, excerises	Quizz, labs, projects, exams

Code	Course Learning Outcomes	Code of PLOs aligned with the program	Teaching Strategies	Assessment Methods
1.2	Recognize the limitations and sources of errors in numerical methods.	K2	Lecture, excerises	Quizz, labs, projects, exams
1.3	Gain introductory exposure to advanced topics like parallel computing,	K1	Lecture, excerises	Quizz, labs, projects, exams
1.4	Understand emerging trends, tools, and applications in scientific computing, including machine learning,	K3	Lecture, excerises	Quizz, labs, projects, exams
2.0	Skills			
2.1	Use programming languages such as Python and MATLAB effectively for scientific computation.	S1	Lecture, excerises	Quizz, labs, projects, exams
2.2	Leverage specialized libraries and frameworks like NumPy, SciPy, and Matplotlib for implementing computational algorithms.	S1	Lecture, excerises	Quizz, labs, projects, exams
2.3	Apply Numerical Methods to Solve Practical Problems	S2	Lecture, excerises	Quizz, labs, projects, exams
3.0	**Values, autonomy, and responsibility**			
3.1	Work in teams to design, develop, and present a computational project that applies scientific computing methods to solve an engineering or science-based problem.	V2	Lecture, excerises	Quizz, labs, projects, exams

C. Course Content

No	List of Topics	Contact Hours
1.	Introduction to Scientific Computing • Topics: Overview of scientific computing, its importance in engineering and science, and comparison with traditional computing.	**3**

2.	Numerical Precision and Errors • Topics: Floating-point arithmetic, truncation errors, and error propagation.	**3**
3.	Linear Algebra in Scientific Computing • Topics: Systems of linear equations, matrix operations, eigenvalues, and eigenvectors.	**6**
4.	Interpolation and Curve Fitting • Topics: Polynomial interpolation, splines, and least-squares fitting.	**6**
5.	Numerical Differentiation and Integration • Topics: Finite difference methods, Newton-Cotes formulas, and Gaussian quadrature.	**6**
6.	Solving Nonlinear Equations • Topics: Root-finding methods (bisection, Newton-Raphson, secant methods). Optimization Techniques • Topics: Unconstrained and constrained optimization, gradient descent, and evolutionary algorithms.	**6**
7.	Scientific Visualization • Topics: Data visualization for scientific data, 2D and 3D plotting.	**3**
8.	Parallel and High-Performance Computing • Topics: Parallel computing with multiprocessing, GPU computing with CUDA or PyTorch.	**6**
9.	Applications in Machine Learning • Topics: Basics of scientific computing in ML, matrix-based optimizations, and neural networks.	**6**
Total		**45**

D. Students Assessment Activities

No	Assessment Activities *	Assessment timing (in week no)	Percentage of Total Assessment Score
1.	**Quizzes**	**2-14**	**15**
2.	**Projects**	**2-14**	**15**
3.	**Assignments**	**2-14**	**10**
4.	**Mid Term**	**7**	**20**
5.	**Final Exam**	**16-17**	**40**

*Assessment Activities (i.e., Written test, oral test, oral presentation, group project, essay, etc.).

E. Learning Resources and Facilities

1. References and Learning Resources

Essential References	• Heath, M. T. (2018). *Scientific computing: An introductory survey* (2nd ed.). SIAM. ISBN 978-1611975574.
Supportive References	• Langtangen, H. P. (2016). *A primer on scientific programming with Python* (6th ed.). Springer. ISBN 978-3662498866. • Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). *Numerical recipes: The art of scientific computing* (3rd ed.). Cambridge University Press. ISBN 978-0521880688.
Electronic Materials	
Other Learning Materials	

2. Required Facilities and equipment

Items	Resources
facilities (Classrooms, laboratories, exhibition rooms, simulation rooms, etc.)	Traditional Classroom
Technology equipment (projector, smart board, software)	Multimedia Projector
Other equipment (depending on the nature of the specialty)	N/A

F. Assessment of Course Quality

Assessment Areas/Issues	Assessor	Assessment Methods
Effectiveness of teaching	Students	Direct, Indirect
Effectiveness of Students assessment	Faculty, Peer reviewer	Direct, Indirect
Quality of learning resources	Faculty, Course coordinator	Direct, Indirect
The extent to which CLOs have been achieved	Course coordinator, Program management committee	Direct
Other		

Assessors (Students, Faculty, Program Leaders, Peer Reviewers, Others (specify)

Assessment Methods (Direct, Indirect)

G. Specification Approval

COUNCIL /COMMITTEE	SOFTWARE ENGINEERING DEPARTMENT COUNCIL
REFERENCE NO.	THE 17TH MEETING FOR THE ACADEMIC YEAR 1446H
DATE	22/04/2025