

COURSE DESCRIPTION

Department and Course Number	CMPS 352	Course Coordinator	Andrew C. Lee
Course Title	Scientific Computing	Total Credits	3
URL		Semester	Fall Semester Only;
		hours	lectures 3 hours per week for 14 week

Current Bulletin Description

Software tools and algorithmic methods for solving large-scale numerical problems in applied science, engineering and real-life applications. Float point and matrix computations, numerical integration and differentiation; Numerical methods for graphics, visualization and game development. Pre-requisite: CMPS 341 with a grade C or better.

Textbook

Cleve B. Moler. Numerical Computing with MATLAB. SIAM 2004 ISBN: 0-89871-560-1. Downloadable from the book's website: <http://www.mathworks.com/moler/chapters.html>

References

See the links as posted on the Moodle site.

Course Goals

- Understand the fundamental computing techniques used in engineering, applied mathematics, physical and life sciences
- Gain the basic skills for using modern scientific computing software (e.g. MATLAB) which is vital in applications such as visualization, computer graphics and game development.

Course Outcomes

Upon successful completion of this course, students will have the following experience.

1. Explore the capabilities of the software tools in visualizing the data. Gain experience in using these tools to help analyze problems empirically.
2. Gain experience in the study of numerical algorithms. Understand issues on correctness, accuracy and efficiency of numerical algorithms.
3. Gain experience in implementing numerical algorithms via MATLAB scripts.
4. Gain experience in developing a term project. It involves the development of a project proposal, study of articles designed for undergraduate students and carrying out implementation and relevant empirical studies based on the required readings. It also involves writing of a technical report to present the findings.

Prerequisites by Topic

1. Basic mathematical background (e.g. sequences, mathematical induction)
2. Basic notions on the analysis of algorithms (e.g. Big-O notation etc.)
3. Experience in developing software solutions for relative complex problems

Major Topics Covered in the Course

1. Review of basic mathematical concepts and techniques (1 week)
2. Floating point computations (1 week)
3. Introduction to vectors and matrices (1 week)
4. Solving non-linear equations (2 week)
5. Matrix Computations I: Solving Systems of Linear Equations (3 week)
6. Numerical Quadrature and Differentiation (1 week)
7. Interpolation, Splines and their applications (2 week)
8. Matrix Computations II: Selected advanced methods (e.g. QR factorization) and their applications (2 week)
9. Advanced topics: Scientific visualization (1 week)

Laboratory projects (specify number of weeks on each)

The number of laboratory assignments is about 8. The major topics are first presented in lectures. During the lab session, students use MATLAB to test the concepts learned and further their understanding of the course materials. Typically, the lab includes an in class demonstration followed by a hands on session. There are homework problems (mostly MATLAB related) stated at the end of most lab exercise

Oral and Written Communications

The student needs to communicate effectively in both writing and oral presentation.

Written: The class participation is assessed by examination on their classnotes. Each student is required to be the note taker for one lecture. Their notes are then revised and posted on our class webpage for other students as a reference. The major set of exercises are MATLAB exercises. We also assign some written homework when appropriate. Each student is required to submit a final report together with the computer implementation (on a separate disk). It is graded as the Final.

Oral: Each student is required to meet with the instructor at least two times to discuss the final topics and present his/her progress.

Social and Ethical Issues

N/A

Theoretical Content

This course emphasizes the use of software tools (e.g MATLAB) and algorithmic methods (e.g. algorithms for Gaussian elimination and LU decomposition etc.). We emphasize the understanding of the theoretical ideas and its importance in the implementation process.