Math 6390, Fall 2013
Topics in Mathematics - Level 6
Scientific Computing

Course Information

89193  Math 6390.001  MW 4:00pm-5:15pm  CB3 1.310

Professor Contact Information

Instructor:       John Zweck
Office:          FO 3.704J
Email:           zweck@utdallas.edu
Webpage:         I will maintain a web page for the course, linked from my web page http://www.utdallas.edu/~zweck. Bookmark it! I will also communicate with you using a class email list. (I do not use eLearning.)
Phone:           (972) 883-6699 (Do not leave a message. Email me instead.)
Office Hours:    W 2:30-3:45pm and by appointment. If you cannot come to my office hours please contact me in class or by email to set up a time to meet. Also, you are encouraged to ask me questions by email.

Course Pre-requisites

The recommended prerequisites are the undergraduate courses MATH 4334 (Numerical Analysis), MATH 4362 (Partial Differential Equations), MATH 3311 (Abstract Algebra I), and the graduate course MATH 6315 (Ordinary Differential Equations). Students without all these prerequisites should consult the instructor to determine whether they have sufficient background to succeed in the course.

Course Description

The course will provide students with a graduate-level introduction to scientific computing. “Computation is now regarded as an equal and indispensable partner, along with theory and experiment, in the advance of scientific knowledge and engineering practice.”

Computational Science is an inherently interdisciplinary activity. Computational scientists make “intelligent use of mathematical software to analyze mathematical models”, and to study

---

1 Direct quote from Graduate Education for Computational Science and Engineering [GECSE], by the SIAM Working Group on CSE Education.
2 [OL, 2.1]
“complex systems and natural phenomena that would be too expensive or dangerous, or even impossible, to study by direct experimentation.”

Paraphrasing and expanding on [OL, 2.1], in this course we will use science and engineering to formulate problems, mathematics and statistics to design mathematical models to study those problems, numerical analysis to design and analyze algorithms to compute solutions, and ideas from computer science to develop (and debug!) mathematical software. Finally, we will close the loop by using visualization and other analytical tools to interpret the results and relate them to the original science/engineering problem. One of our major goals is to answer questions about the accuracy of the computed solution (code verification) and the accuracy and range of applicability of the mathematical model (model validation).

The course will be organized around several projects designed to illustrate the process of scientific discovery using computation, and to give students formative experiences doing computational science with prototypical problems. The projects will be done in MATLAB and Python. Lectures will provide students with the necessary background required to complete the projects including physics-based mathematical modeling, numerical analysis, and the acquisition of sufficient expertise with both MATLAB and Python. The projects are:

- **The Three Body Problem**: A model of the gravitational interaction between three bodies will be used to study the nonlinear dynamics of a smaller mass in the gravitational field of two larger masses. The project will involve numerical solution of systems of ODEs, finding stationary solutions and analyzing their stability (using nonlinear solvers and unconstrained numerical optimization), demonstrating chaotic behavior, and searching for closed orbits. (Programming in MATLAB.)

- **Monte Carlo Simulation**: In this project students will learn how to generate random numbers, to sample from statistical distributions, and to use Monte Carlo simulations to approximate high-dimensional integrals (such as arise in thermodynamics). (Programming in MATLAB.)

- **The Dynamics of Noise Processes in Soliton Systems**: The propagation of light in optical fibers is governed by a generalization of the nonlinear Schrödinger equation (NLSE). The NLSE is a partial differential equation that has special solutions called solitons which are pulses of light that retain their shape as they propagate, even during collisions. The goal of this project is to numerically study the collision dynamics of solitons and to use Monte Carlo simulations to study the effects that optical noise has on the propagation of individual solitons. The ideas developed in this project have applications to the design of femto-second lasers and optical fiber communications systems. (Programming in MATLAB.)
• **Symbolic Computation to explore group theoretic properties of the Rubik’s Cube puzzle:** Symbolic Computation software such as SAGE enables the user to perform large algebraic calculations that would otherwise be intractable. In this project students will obtain an introduction to symbolic computation in the context of the group of symmetries of the Rubik’s Cube puzzle. (Programming in Python.)

• **Finite Element Solution of PDEs:** In this project students will use the mathematical software FEniCS to solve partial differential equations by formulating them as variational problems and applying the finite element method. Each student’s project will be individually tailored based on his/her prior experience with PDEs and their research interests. (Programming in Python.)

**Required Textbooks and Materials**

There is no required text. The recommended textbook for the course is

[OL] “Scientific Computing with Case Studies”, by Diane O’Leary, SIAM 2009. (*This book is close in spirit and in some of the content to our course.*)

The following books, e-books, and online resources may prove helpful. I will use them to prepare course materials. You are *not* expected to purchase these materials. Some are freely available, others will be placed on reserve in the library.

**Scientific and Symbolic Computing**

[NK] “Data-Driven Modeling and Scientific Computing”, by N. Kutz, forthcoming from Oxford University Press, Fall 2103. (*Two of the course projects will be adapted from this book.*)


**Software (and Documentation)**

[FEniCS] FEniCS Project for automated, efficient solution of differential equations. (*FEniCS has a user-friendly Python scripting interface.*)

[SAGE] SAGE is a compilation of existing open-source packages. (*SAGE has a Python-based interface. We will use SAGE for the symbolic computation project.*)
Background Texts in Numerical Analysis and Statistics


[DL] “Online Statistics Education: An Interactive Multimedia Course of Study (Version 2.0)”, by D. Lane et al., Rice University.

MATLAB

[MW] Mathworks MATLAB Documentation. (Everything is here if you know where to look!)


Python

[PD] Python Documentation, including a tutorial.


Academic Calendar and Assignments

Projects will be assigned for each component of the course and will be posted, together with their due dates, on the course web page. The course schedule, lecture notes, and other resources will also be posted on the course web page as they become available.
Grading Policy

Course grades will be based entirely on projects. There will be no midterm exam and no final exam. Because some projects may be more involved than others, the projects may not all be worth the same number of points. The total number of points each project is worth will be stated on the project assignment, together with a breakdown of the number of points allocated for each part of the project.

How I assign final grades

For each graded project I work out how many points I expect a student who has a solid understanding of the material to get. I tend to put the bottom B near this score. Then I work out where to place the bottom A,C,D by looking at individual projects. Then I take an imaginary student who got the bottom B (say) for each project and calculate their score. If your score is higher than the imaginary student’s you get a B. For borderline students, I will go back and carefully look at the performance on each project.

Instructor Policies

Projects

You may discuss issues related to the projects with other students in the class. However you are expected to write and submit your own code, results, and reports. You may not use code or results from the web unless authorized by the course instructor. However, because of the interdisciplinary nature of the subject, you will frequently need to learn background material and do fact checking for yourself. Any background material that you rely on should be carefully referenced. For material from a book give the author, book title, and chapter. For journal articles give author, title, journal, issue, page numbers, year and URL (if you obtained it from the web). For other material from the web, give URL, author and institution, if known.

Academic Integrity

I will be vigorous in reporting all instances of cheating, including plagiarism, to the University administration. See http://www.utdallas.edu/deanofstudents/dishonesty/

UT Dallas Syllabus Policies and Procedures

The information at http://go.utdallas.edu/syllabus-policies constitutes the University’s policy and procedures segment of the course syllabus.

The descriptions and timelines contained in this syllabus are subject to change at the discretion of the Professor.