PSCB57

Introduction to Scientific Computing

Dr. Brian Wilson
Office: SW-506G
brian.wilson@utoronto.ca

COURSE DESCRIPTION:

This course is an introduction to the use of computers in the physical and biological sciences emphasizing the choice and design of algorithms and their implementation in a high-level computer language for the solution of problems arising in the physical and biological sciences. Topics will include elementary numerical analysis, such as numerical integration, mathematical modeling of physical systems, data fitting and interpolation. (Intended primarily for physical and biological science students who do not plan to pursue any of the Programs in computer science or cognitive science.)

COURSE OBJECTIVES

By the end of this course you should be able to find approximate answers to many common mathematical problems which do not have exact answers. For example, you will be able to write software which would create your own scientific calculator, including all the trigonometric functions, using only basic mathematical operators such as addition, subtraction, multiplication and division. You will also be able to find approximate values of definite integrals, approximate solutions to ordinary differential equations with specific initial conditions, and interpolate graphs given a discrete set of data points.

You need to know basic calculus including derivatives, integrals and Taylor series. You do not need to have taken a course on ordinary differential equations or any other mathematics. While you do not need to know how to program Python, you will be expected to learn the basics of programming very quickly. A substantial amount of self-guided learning of Python skills is required to do well in this course.

TEXTBOOK:

None. Check

`http://pathfinder.scar.utoronto.ca/~dyer/csca57/book_P/book.html`

for the original course notes for this class, written by Professor Charles Dyer. The course is still heavily based on these notes, so they are a good reference material.

OFFICE HOURS:

TBD. Check Blackboard for the days/times. I will try to reply to emailed questions with 48 hours. If I do not, please re-send the email so that it pops up at the top of my in-box.

LECTURE NOTES:

Lecture notes (i.e., slides) will be posted on Blackboard.

ASSESSMENT:

FIRST MIDTERM:	25%
SECOND MIDTERM:	25%
FINAL EXAM:	50%

If you feel you did poorly on a test, you can talk with me during my office hours and then you can move up to 10% (of the total 25%) of that test weight to your final exam. For example, if you do poorly on the first test then it would be worth 15%, the second would be 25% as usual, and your exam would be 60%. There is a deadline for making this decision which will be announced. The deadline will be roughly one to two weeks after the tests are returned in tutorials. You can do this for both tests, in which case your final exam will be worth 70% of your grade.

TEST AND EXAM:

The first test should be in early February. The second test should be in mid March. The second test will only cover material that was not tested on the first test.

The final exam will cover **all** material.

If you miss a midterm for a valid excuse (like a medical issue), your midterm grade will be added to your final exam. Thus your final exam would be worth 75%. If you miss both midterms you need to talk with me because having a final exam worth 100% of your grade is worrisome.

TUTORIALS:

Students registered in the course are expected to enroll in one tutorial section. Tutorials are 50 minutes in duration, and are held every Thursday starting in the second week of class. Tutorials will typically focus on the practical details of implementing the theory learned in lectures. As such, they will provide the primary opportunity for guided learning with regards to Python. Each tutorial will end with a quick quiz to help you assess how well you understand that week's material.

ASSIGNMENTS AND TUTORIAL QUIZZES:

I will give assignments and tutorial quizzes. These are optional in the sense that they do not give you grades. However, if you do not attempt them you will likely do poorly on the tests and exam, so I will assume everyone attempts them. I strongly encourage you to use them as learning opportunities. The best way to do well in the course will be to learn the material; memorization will not get you far in this course. The best way to learn the material is to struggle with writing programs that solve the assignments.

TENTATIVE **LECTURE SCHEDULE**

Week 1
Introduction to Programming

Week 2
How Computers Work

Week 3
Calculus: Taylor Series

Week 4
Interpolation and Extrapolation

Week 5
Finding Roots of Functions

Week 6
Numerical Integration: Basic

Week 7
Numerical Integration: Gaussian Quadrature and Adaptive Methods

Week 8
Differential Equations: Basic

Week 9
Differential Equations: Advanced (Runge-Kutta Methods)

Week 10
Differential Equations: Adaptive Methods

Week 11
Monte Carlo: Basics and Integration

Week 12
Monte Carlo: Select Applications

I usually find that we go slightly slower than the above outline indicates. We might not do as much Monte Carlo as the outline predicts.