Introductory Computational Physics

This course is intended to introduce main computational tools, techniques and methods used in physics. Student will practice writing, compiling, and running computer programs, together with analysis of results, and presentation of their results as scientific reports.

This is not a course in computing science, or in programming. It focuses specifically on practical methods for solving physics problems. The course is therefore designed such that a significant fraction of the students' time is spent actually programming specific physical problems.
This course is **learning by creating**.

University Catalog

420/520 Introductory Computational Physics. Credit hours 3.
Introduction to computationally based problem solving in physics with an emphasis on understanding and applying various numerical algorithms to different types of physics problems. Topics will include numerical integration (quadrature), numerical solution of ordinary differential equations (Runge-Kutta and Numerov methods), polynomial approximations, numerical linear algebra, and Monte-Carlo methods. These computational methods will be applied to problems in classical and quantum mechanics, as well as electromagnetic theory.

Prerequisite

Phys 319, Phys 323, Math 212, CS 150 or CS 153.
Students are expected to be familiar with at least one programming language, such as C, C++, Python, MatLab, Java, Fortran.

Requisites

Hardware and software essential for this class: a computer with a language/software of **your** choice.

Course Structure

This course is conducted **asynchronously,** allowing students to complete the modules and assignments at their own pace within the weekly schedule. Each week consists, on average, of the following components: 1) Two video lectures accompanied by lecture notes (as reading material), 2) An assignment (either as exercises or a project), 3) Office hours as Zoom meetings. At the beginning of each week students will have a detailed schedule for the week.

Course materials

Class materials will be published on Canvas's class page (see *Modules* for lecture notes, and *Media Gallery* for video lectures)

Instructor

Dr. Alexander Godunov
Office: OCNPS 0219 (Oceanography and Physics)
Phone: 683-5805
email: agodunov@odu.edu
Web: https://www.agodunov.com

Textbooks

There is no book that could serve as a sole book for the course. However, there are many excellent books (see page 5 of the syllabus).

Support resources

Office hours: Tuesday, Friday. Time 15:00-16:00
Place: Zoom https://odu.zoom.us/j/93371650052
Meeting ID: 933 7165 0052, Passcode: CPhys420F4
and by appointment (the same Zoom meeting info).
E-mail: agodunov@odu.edu (have "Physics 420" in the subject line).

Course grades

The final grade is calculated on an absolute scale. There are 100 points possible for this course of which
40 points are for homework assignments,
40 points for midterm projects,
20 points for the final project.
A letter grade is determined only at the end of the term.
Grade Requirements

92 ≤ A < 100	88 ≤ A- < 92	83 ≤ B+ < 88	
75 ≤ B < 83	70 ≤ B- < 75	65 ≤ C+ < 70	
60 ≤ C < 65	55 ≤ C- < 60	50 ≤ D < 55	F < 50

Homework

Homework assignments will be set as we progress through the course. One assignment will be due approximately each week. All assignments should be submitted electronically. Doing homework problems is one of the best ways to learn the material. You should start homework early and get help if needed before the due date.
Filename standard: LastName_HXX.pdf where XX in a homework assignment.
No individual extensions of assignment submission dates will be given.

Projects

There will be midterm computational projects and one final project. The projects will aim to solve specific problems in physics. You will be required to submit a report on each project. All reports should have the following sections: title, description of the problem, equations and computational model, testing, example of input parameters, results (figures, tables, analysis), and conclusion.

The report and your computer program should be submitted electronically by the due date and time (as an attachment, including an example with input/output files). This will enable the assessors to check and run your program if necessary. Please, remember that "no submitted program = no credit for the assignment". Recommended format for reports is PDF, with computer codes attached as txt files. Each assignment will be graded for completeness, correctness and clarity. No make-up projects will be given.
Filename standard: LastName_PXX.pdf where XX in an assigned project.

Final Exam/Project

Since this is a "learn by doing" course, there will be no traditional final exam but a final project. The final project should be written as a short research paper based on your calculations.

Keys to success

Right motivation, working diligently, effectively and efficiently is the key to success. If you work regularly and allocate enough time each day to practice and complete the assignments on time and keep up with the course, you will get the most out of the course both intellectually and grade-wise.

You should invest about 6-9 hours per week outside of class to succeed in this course. This is consistent with university guidelines (i.e. two to three hours of outside preparation time for every credit hour). Students with little or no programming experience (or those who love the subject and wish to do extremely well in it) may want to put in more hours..

You are recommended to start your assignments well before the last night when your assignments are due. It is a general experience that a computer program usually does not work correctly (if works at all) at the beginning. A search for a problem, or a computer bug, may take more time that you expect.

Collaboration

Collaboration is strongly encouraged. Because the course is graded on an absolute scale, you will never reduce your grade by helping others — on the contrary, by doing so you will reinforce your own knowledge and improve your performance. Although, before working together or consulting others on any assignments, it is helpful to first tackle the work alone.

Activities for which collaboration is encouraged: group projects, understanding issues discussed in class, using compilers and other software, using numerical libraries.

Activities for which collaboration is NOT permitted are homework assignments, individual projects and the final examination.

Professional Integrity

In Physics 420/520 high professional and ethical standards are promoted. Plagiarism and cheating are serious offenses and may be punished by failure on the exam and failure in the course. The academic integrity code is to be maintained at all times. Using Google, Chegg, and similar resources for getting solutions to homework problems and projects is considered as cheating.

Accommodation

Students are encouraged to self-disclose disabilities that have been verified by the Office of Educational Accessibility by providing Accommodation Letters to their instructors early in the semester in order to start receiving accommodations. Accommodations will not be made until the Accommodation Letters are provided to instructors each semester.

Using AI Tools

In this course, we encourage the exploration of AI tools to enrich your learning experience. These innovative tools can offer valuable assistance. However, it is important to note that these tools should be used responsibly and ethically. While AI can be a powerful aid, it is crucial to understand that using AI tools to cheat on assignments is strictly prohibited. The purpose of assignments is to assess your understanding and skills; cheating not only undermines this process but also diminishes your own growth. Therefore, we expect all students to use AI tools as a supplement to their learning, but not to cheat or gain an unfair advantage.
The use of AI tools as automated code generators is strictly forbidden.

Course Outline

Part 1: Introduction to computing

1. Tools of computational physics
2. Numerical and physics libraries
3. Computer languages (C/C++, MatLab, Fortran, Python, …) and physics
4. ** High-performance computing

Part 2: Introduction to basic numerical methods with applications to physics

1. Interpolation.
2. Derivatives
3. Numerical integration
4. Solution of non-linear equations
5. Ordinary differential equations: initial and boundary value problems
6. Linear Algebra: systems of linear equations, eigenvalue problems
7. * Data fitting and analysis
8. ** Partial Differential Equations (elliptic, parabolic, hyperbolic)

Part 3: Application to physics, computer simulation methods

1. Classical mechanics
    - 1.1 Motion of projectiles, satellites, and planets
    - 1.2 Oscillatory motion, including chaotic motion
    - 1.3 Classical scattering
    - 1.4 ** Molecular dynamics simulation
2. Quantum mechanics
    - 2.1 Time-independent Schrodinger equation
    - 2.2 * Time-dependent Schrodinger equation
    - 2.3 ** Quantum scattering
3. Monte Carlo simulation
    - 3.1 Random number generators and distributions
    - 3.2 Monte Carlo integration (including Metropolis sampling)
    - 3.3 Random walk with applications
    - 3.4 Some application of Monte Carlo simulation to classical and quantum systems
4. Electromagnetic theory
    - 4.1 * Potentials and fields
5. Additional chapters
    - 5.1 * Based on available time and class interest

Some topics may be added, and some topics omitted at the instructor's discretion, and depending on student's interest and time constraints.

* *Depending on class preparedness*

** *Indicates "optional" sections* (if we get that far)

Books

Computational Physics (recommended for the class)

1. R. Landau et. all
    a. Computational Physics. Problem solving with computers (Wiley 1997)
    First edition: **C++ and Fortran** based
    b. Computational Physics. Problem solving with computers (Wiley 2007)
    Second edition: **Java** based
    c. Computational Physics. Problem solving with computers (Wiley 2015)
    Third edition: **Python** based
    d. Computational Problems for Physics:
    With Guided Solutions Using **Python** (CRC Press 2018)
2. Jay Wang, Computational Modeling and Visualization of Physical Systems with **Python** (Wiley 2015)
3. Mark Newman, Computational Physics (Great Space 2012) (with examples in **Python**)
4. H. Gould et all, An Introduction to Computer Simulation Methods: Applications to Physical Systems, (Addison-Wesley; 3rd edition 2006)
   Java based
5. Jos Thijssen, Computational Physics (Cambridge University Press; 2nd edition 2007)
   Upper-level graduate text with an emphasis on condensed matter applications.
   Language independent text

Numerical methods

1. J.D. Hoffman, Numerical Methods for Engineers and Scientists (CRC Press; 2nd edition 2001)
   with **Fortran** examples
2. T. Sauer, Numerical Analysis (Pearson, 3rd edition, 2018) [with **MatLab** examples]
3. W.H.Press et all, Numerical Recipes. The Art of Scientific Computing (Cambridge University Press 3rd edition 2007)
   C++ based, note that other editions are centered on **Fortran, C, Pascal**
4. J.R.Dormand, Numerical Methods for Differential Equations: A Computational Approach (CRC-Press 1996)
   with **Fortran** examples
5. P. Davis and P. Rabinowitz, Methods of Numerical Integration (Academic Press 1984)
   with **Fortran** examples

Computer languages (many excellent books, here my favorites)

1. P. Deitel and H. Deitel, **C++** How to program (Pearson; 10th edition 2016)
2. B. Hahn and D.T. Valentine, Essential **MATLAB** for Engineers and Scientists (Academic Press; 7th edition 2019)
3. M. Lutz, Learning **Python** (O'Reilly, 5th Edition Fifth Edition 2013)
4. S. Chapman, **Fortran** for scientists and engineers (McGraw Hill; 4th edition 2017)