Old Dominion University Physics 420/520 Spring 2010

Introductory Computational Physics

This course is intended to give an introduction to main computational tools, techniques and methods used in contemporary 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. In this course the students should learn by doing.

University Catalog
420/520 Introductory Computational Physics. CRN 23319.
Credit hours 3.
Introduction of computational methods and visualization techniques for problem solving in physics.

Prerequisite
Students are expected to be familiar with at least one programming language (Fortran, C, C++, Basic, or Java).
Physics 231/232 and Math 211/212

Instructor
Dr. Alexander Godunov
Office: OCNPS 0219 (Oceanography and Physics)
Phone: 683-5805
email: agodunov [at] odu.edu
The course web page: http://www.odu.edu/~agodunov/teaching/phys420_10

Classes
Monday, Wednesday 13:30 – 14:45
Oceanography & Physics, Room 303

Office Hours
Monday, Wednesday 14:50 – 16:00
Oceanography & Physics, Room 219 (office)
and by appointment

Materials (recommended)
There is no a single book that could serve as a textbook for the course. Therefore most material is presented in the lecture notes, available from the web page - http://www.odu.edu/~agodunov/teaching/notes/index.html.
A list of helpful books can be found on the course webpage (see textbooks)

Course structure
Different people learn in different ways. Therefore this course offers a learning environment with a diversified set of options that you can tailor to your individual learning style. There are class meetings, homework assignments, computational projects, and practice. There are many ways to get assistance with the material in this course: the office hours, appointments, e-mail, phone

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. Please, let me know as soon as possible if you have difficulties and may need extra help.

You are recommended to start you 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.

Homework

Doing the homework problems is one of the best ways to learn the art of computational physics. Credit for homework is given to encourage practicing and thinking about computational physics on a regular basis. This credit influences the final grade for this course. Assignments may be submitted early. There will be a penalty for late submissions (see grades).

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 formats for reports are MSWord and PDF, with all figures embedded in the body of the document. A carbon copy of your report should be in the instructor’s mailbox as well.

Each assignment will be graded for completeness and correctness. Clarity of your presentation may affect your grade.

No make-up projects will be given. In case you have a legitimate reason for missing a project deadline, consult with me before, or within 24 hours after the deadline.

Regrade: Requests for correction of grading mistakes on projects can be made when the work is returned to you. The requests must be made within three days after getting your grade. Regrade requests should be written. In their request, students must explain why they believe there is a mistake in grading and why they deserve more credit. It is not a plea for more points

Exams

Since this is a “learn by doing” course, there will be no traditional exams but midterm projects, and one final project. The final exam will consist of an oral presentation of the final project and an accompanying written paper (in the format of a research article).

The final examination is mandatory and will be given only at the scheduled time –

May 3, 2010 (Monday) at 12:30.

Grades

The final grade is calculated on an absolute scale. There are 100 points possible for this course of which
30 points are for homework assignments,
50 points for midterm projects,
20 points for the final project and examination.
A letter grade is determined only at the end of the term.
Grade forgiveness policy: assignments (homework, projects) handed in late without an excuse will be pro-rated as followed: 90% up to 24 hours late; 80% up to 48 hours late; 50% thereafter until the last day of class.

Attendance

Attendance in lectures is optional; however, you are responsible for all information discussed in class, including any announcements.

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.

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 Conduct

In Physics 420/520 high professional standards, including ethical standards, are promoted. Plagiarism and cheating are serious offenses and may be punished by failure in the course. The academic integrity code is to be maintained at all times.

Course Outline

Part 1: Introduction to computing

1. Introduction (What is computational physics?)
2. Tools of computational physics
3. Short introduction to C/C++
4. **Short introduction to Fortran 90/95*

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

1. Interpolation: linear and polynomial interpolation, divided difference polynomials, equidistant points - Newton's forward/backward difference, spline interpolation.
2. Derivatives: Lagrange polynomials, Newton difference polynomials, finite difference approximations.
3. Numerical integration: simple quadratures (trapezoid, Simpson), Newton-Cotes formulas using divided difference polynomials, Gauss quadratures, integration with adaptive step size, special cases (oscillating functions, improper integrals, singularities, multiple integrals)
4. Solution of non-linear equations: closed domain methods (bisection and regula falsi), open domain methods (the Newton's method, the secant method), nonlinear systems of equations (the Newton's method)
5. Ordinary differential equations
   - *5.1* Initial value problems: the first-order Euler method, the second-order single point methods (predictor-corrector, leap-frog, Störmer-Verlet), Runge-Kutta methods, adaptive methods.
   - 5.2 boundary value problems: the shooting method, the equilibrium method, the Numerov's method, the eigenvalue problems - the equilibrium method
6. Linear Algebra (systems of linear equations, the eigenvalue problem)
   - 6.1 Direct elimination methods: simple direct elimination, Gauss elimination (pivoting, scaling), inverse matrices using LU Doolittle factorization, determinants, tri-diagonal systems (Thomas method), ill-conditioned systems and condition numbers.
   - 6.2 Iterative methods: Jacobi iteration, Gauss-Seidel, successive over-relaxation (SOR).
   - 6.3 Eigenvalue problem: the direct power methods, Jacobi method for symmetric matrices, the QR method.
7. ** Fourier transforms: discrete Fourier transforms, fast Fourier transforms*

Part 3: Application to physics & Computer simulation methods

1. Classical mechanics
   - 1.1 Projectile and particle motion (including physics of sport)
   - 1.2 Few body problems in application to satellite and planetary motion.
   - 1.3 **Classical scattering*
2. Monte Carlo simulation
   - 2.1 Random numbers (generators, uniform and non-uniform distributions),
   - 2.2 Monte Carlo integration,
   - 2.3 Metropolis sampling
   - 2.4 Random walk (unrestricted, restricted, persistent, self-avoiding), diffusion.
   - 2.5 Introduction to the Monte Carlo simulations for the many-body problem.
3. Quantum Mechanics

   3.1 1D Schrödinger time-independent equation (wave functions and eigenvalues) (the Numerov's method, the Monte-Carlo variational method, the equilibrium method for the ODE)

   3.2 *****_1D time-dependent Schrödinger equation_

4. Chaos: Poincare maps, the butterfly effect, **Fourier analysis of nonlinear systems*
5. Data fitting and analysis: least squares method, linear and non-linear functions
6. Introduction to molecular dynamic simulation
7. **Cellular automata (forest fire)*

Part 4. Software, libraries, computational clusters.

1. Introduction to scientific software (Maple, Mathematica)
2. Plotting, visualization and animation software.
3. Libraries of numerical methods
4. Libraries of physics programs
5. High-performance computing

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

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