Computational Physics Lectures: Introduction to the course

Morten Hjorth-Jensen[1,2]

Department of Physics, University of Oslo[1]

Department of Physics and Astronomy and National Superconducting Cyclotron Laboratory, Michigan State University[2]

Aug 4, 2018

Overview of first week

- Thursday: First lecture: Presentation of the course, aims and content
- Thursday: Second Lecture: Introduction to C++ programming (chapters 2 and 3 of lecture notes) and start discussion of project 1.
- Friday: Numerical precision and C++ programming, continued and discussion of project 1 (chapter 2 and 3 of lecture notes)
- Numerical differentiation and loss of numerical precision (chapter 3 lecture notes)
- Computer lab: Thursday and Friday. First time: Thursday and Friday this week, Presentation of hardware and software at room FV329 first hour of every labgroup and solution of first simple exercises. The first two weeks we focus on simple programming exercises and to set up github and QTcreator. This week we discuss how to set up git and obtain a github account and look at two simple programming exercises and for those interested start with project 1.

Reading suggestions and exercises

- Read sections 2.1-2.5 and 3.1-3.2 of lecture notes:
  - Introduction to C++ programming
  - Numerical precision and C++ programming (chapter 2 of lecture notes)
  - Numerical differentiation and loss of numerical precision (chapter 3 lecture notes)

Lectures and ComputerLab

- Lectures: Thursday (8.15am-10am) and Friday (8.15am-10am).
- Weekly reading assignments needed to solve projects.
- First hour of each lab session may be used to discuss technicalities, address questions etc linked with projects.
- Detailed lecture notes, exercises, all programs presented, projects etc can be found at the homepage of the course.
- Computerlab: Thursday (10am-6pm) and Friday (10am-6pm) room FV329.
- Weekly plans and all other information are on the official webpage.
- No final exam, the last three projects are graded. In total five projects which all have to be approved.

Course Format

- Five compulsory projects. Electronic reports only using devilry to hand in projects and Git for repository and all your material.
- Evaluation and grading: The last three projects are graded and each counts 1/3 of the final mark. No final written or oral exam.
- The computer lab (room FV329)consists of 16 Linux PCs, but many prefer own laptops. C/C++ is the default programming language, but Fortran2008 and Python are also used. All source codes discussed during the lectures can be found at the webpage and github address of the course. We recommend either C/C++, Fortran2008 or Python as languages.

Teachers and ComputerLab

Teachers :

1. Anna Gribovskaya
2. Morten Hjorth-Jensen
3. Anders Johansson
4. Mathias M. Vege
5. Sebastian G. Winther-Larsen

day	teacher
Group 1: Thursday 10am-2pm	Anders, Anna, Mathias, MHJ, Sebastian
Group 2: Thursday 2pm-6pm	Anders, Anna, Mathias, MHJ, Sebastian
Group 3: Friday 10am-2pm	Anders, Anna, Mathias, MHJ, Sebastian
Group 4: Friday 2pm-6pm	Anders, Anna, Mathias, MHJ, Sebastian

Deadlines for projects (end of day)

1. Project 1: September 10 (not graded, only feedback)
2. Project 2: October 1 (not graded, only feedback)
3. Project 3: October 22 (graded with feedback)
4. Project 4: November 12 (graded with feedback)
5. Project 5: December 10 (graded with feedback)

Projects are handed in using devilry.ifi.uio.no. We use Github as repository for codes, benchmark calculations etc. Comments and feedback on projects only via devilry.

Topics covered in this course

- Numerical precision and intro to C++ programming
- Numerical derivation and integration
- Random numbers and Monte Carlo integration
- Monte Carlo methods in statistical physics
- Quantum Monte Carlo methods
- Linear algebra and eigenvalue problems
- Non-linear equations and roots of polynomials
- Ordinary differential equations
- Partial differential equations
- Parallelization of codes
- High-performance computing aspects and optimization of codes

Syllabus

Linear algebra and eigenvalue problems, chapters 6 and 7

- Know Gaussian elimination and LU decomposition
- How to solve linear equations
- How to obtain the inverse and the determinant of a real symmetric matrix
- Cholesky and tridiagonal matrix decomposition

Syllabus

Linear algebra and eigenvalue problems, chapters 6 and 7

- Householder's tridiagonalization technique and finding eigenvalues based on this
- Jacobi's method for finding eigenvalues
- Singular value decomposition
- Cubic Spline interpolation

Syllabus

Numerical integration, standard methods and Monte Carlo methods (chapters 4 and 11)

- Trapezoidal, rectangle and Simpson's rules
- Gaussian quadrature, emphasis on Legendre polynomials, but you need to know about other polynomials as well.
- Brute force Monte Carlo integration
- Random numbers (simplest algo, ran0) and probability distribution functions, expectation values
- Improved Monte Carlo integration and importance sampling.

Syllabus

Monte Carlo methods in physics (chapters 12, 13, and 14)

- Random walks and Markov chains and relation with diffusion equation
- Metropolis algorithm, detailed balance and ergodicity
- Simple spin systems and phase transitions
- Variational Monte Carlo
- How to construct trial wave functions for quantum systems

Syllabus

Ordinary differential equations (chapters 8 and 9)

- Euler's method and improved Euler's method, truncation errors
- Runge Kutta methods, 2nd and 4th order, truncation errors
- How to implement a second-order differential equation, both linear and non-linear. How to make your equations dimensionless.
- Boundary value problems, shooting and matching method (chap 9).

Syllabus

Partial differential equations, chapter 10

- Set up diffusion, Poisson and wave equations up to 2 spatial dimensions and time
- Set up the mathematical model and algorithms for these equations, with boundary and initial conditions. Their stability conditions.
- Explicit, implicit and Crank-Nicolson schemes, and how to solve them. Remember that they result in triangular matrices.
- How to compute the Laplacian in Poisson's equation.
- How to solve the wave equation in one and two dimensions.

Overarching aims of this course

- Develop a critical approach to all steps in a project, which methods are most relevant, which natural laws and physical processes are important. Sort out initial conditions and boundary conditions etc.
- This means to teach you structured scientific computing, learn to structure a project.
- A critical understanding of central mathematical algorithms and methods from numerical analysis. In particular their limits and stability criteria.
- Always try to find good checks of your codes (like solutions on closed form)
- To enable you to develop a critical view on the mathematical model and the physics.

Additional learning outcomes

- has a thorough understanding of how computing is used to solve scientific problems
- knows some central algorithms used in science
- has knowledge of high-performance computing elements: memory usage, vectorization and parallel algorithms
- understands approximation errors and what can go wrong with algorithms
- has experience with programming in a compiled language (Fortran, C, C++)
- has experience with debugging software
- has experience with test frameworks and procedures
- can critically evaluate results and errors
- understands how to increase the efficiency of numerical algorithms and pertinent software
- understands tools to make science reproducible and has a sound ethical approach to scientific problems

Computing knowledge

Our ideal about knowledge on computational science

Hopefully this is not what you will feel towards the end of the semester!

©Zits Partnership

And, there is nothing like a code which gives correct results!!

- J. J. Barton and L. R. Nackman,*Scientific and Engineering C++*, Addison Wesley, 3rd edition 2000.
- B. Stoustrup, *The C++ programming language*, Pearson, 1997.
- H. P. Langtangen INF-VERK3830 `http://heim.ifi.uio.no/~hpl/INF-VERK4830/`
- D. Yang, *C++ and Object-oriented Numeric Computing for*

Other courses in Computational Science at UiO

Bachelor/Master/PhD Courses

- INF-MAT4130 Numerical linear algebra
- MAT-INF4300, PDEs and Sobolev spaces I
- MAT-INF4310, PDEs and Sobolev spaces II
- MAT-INF3360 - Introduction to Partial Differential Equations
- INF5620 Numerical methods for PDEs, finite element method
- FYS4411 Computational physics II (Parallelization (MPI), object orientation, quantum mechanical systems with many interacting particles), spring semester
- FYS4460 Computational physics III (Parallelization (MPI), object orientation, classical statistical physics, simulation of phase transitions, spring semester
- INF3331/4331 Problem solving with high-level languages (Python), fall semester
- INF3380 Parallel computing for problems in the Natural Sciences (mostly PDEs), spring semester

Extremely useful tools, strongly recommended

and discussed at the lab sessions

- GIT for version control (see webpage), this week
- ipython notebook
- QTcreator for editing and mastering computational projects (for C++ codes, see webpage of course), next week
- Armadillo as a useful numerical library for C++, highly recommended, week 36 and rest of semester
- Unit tests, week 37 and later
- Devilry for handing in projects, next week