Applied Mathematics 225

Advanced Scientific Computing: Numerical Methods

Lecturer: Chris H. Rycroft

Logistics

Lectures: Monday/Wednesday, 10:30 AM ET

Email: `chr@seas.harvard.edu`

Course website:
`https://courses.seas.harvard.edu/courses/am225`

Logistics

CHR office hours: Tuesday 10 PM–10:30 PM ET
Wednesday 3 PM–4 PM ET

TFs: Nick Boffi (boffi@g.harvard.edu)
Eder Medina (e_medina@g.harvard.edu)

AM 225: a second course in scientific computing

Harvard already has a long-running course in scientific computing and numerical analysis, Applied Math 205,[1] which is offered in the fall. It covers a broad overview of scientific computing. Students from many disciplines find it useful.

However, many important topics are not covered in AM205 or other Harvard graduate courses, but are at our peer institutions.[2]

This course aims to fill this need, and cover further topics is scientific computing. It will also delve more into implementation issues on modern computer architectures.

---

[1] https://courses.seas.harvard.edu/courses/am205

[2] For example, UC Berkeley Mathematics has three regular graduate courses in numerical methods: Math 221, 228A, 228B.

Courses to compare to

- **Applied Computing 274: Computational Methods in the Physical Sciences.** This course is sometimes lectured by Prof. Sauro Succi in the Fall semester. It covers some similar topics, but it focuses on physical applications. AM225 will be taught from an applied math perspective.
- **Applied Math 202: Physical Mathematics II.** [Not offered this year.] Has a focus on theoretical tools for PDEs. Is a good complement to this course.

Courses to compare to

- **MIT 16.930: Advanced Topics in Numerical Methods for Partial Differential Equations.** This course focuses on the discontinuous Galerkin method, a PDE solution method that is the subject of much contemporary research.
- **Many more courses at MIT in different departments.**
- **NYU: Advanced Topics in Numerical Analysis: High Performance Computing.** This course by Prof. Georg Stadler blends applied mathematics and issues in computational implementation.

Prerequisites

This course will assume that students have already taken a first course in scientific computing that is similar to *Applied Math 205*.

Familiarity with linear algebra, calculus, and partial differential equations will be assumed.

An intermediate level of programming ability (*e.g.* at the level of CS 50/51, AM 111/205) will be assumed. The course will also require the use of either C++ or Fortran—students should either be familiar with this, or have a willingness to learn.

Programming languages

In-class demos will make use of both Python and C++.

Python will be used for conceptual demos.

C++ will be used for demos focusing on performance. We will also make use of the OpenMP library for multithreading.

You can complete many homework problems in any language. Some homework problems will require the use of OpenMP.[3]

[3] OpenMP also runs in Fortran.

Advantages of Python

Why Python?

- Freely available, widely used, and versatile
- Interpreted language, good for small tasks without the need for compilation
- Good linear algebra support via NumPy and SciPy extensions[4]
- Good visualization support via Matplotlib[5]

[4] http://www.numpy.org and http://www.scipy.org/
[5] http://www.matplotlib.org

Advantages of C++

Why C++?

- Freely available, widely used, and versatile
- Compiled language with high performance that provides a better view of what is happening at the hardware level
- Object-oriented syntax makes it well-suited to managing larger tasks
- Can directly link to system and third-party libraries
- Has methods to link to interpreted languages (*e.g.* `mex` for MATLAB, `swig` for Python)

C++ tutorial

We plan to offer several tutorials on C++ in the first few weeks of the course. The first will cover the basics of C++ programming,

and is aimed at people who have experience coding in other languages (*e.g.* MATLAB, Python).

Website and use of Piazza

Main website is at
http://courses.seas.harvard.edu/courses/am225/

We will use Piazza for homework discussion. There is a link to sign up from the website. I will also invite all enrolled students to join.

For any course-related question of general interest, please post to Piazza.

Syllabus (part 1)

0. **Introductory examples and OpenMP programming**
1. **Advanced ODE integration methods and theory**
   1.1 Order conditions
   1.2 Implicit Runge–Kutta schemes, ODE stability, parallel methods
   1.3 Methods for second-order ODEs, symplectic integration methods
   1.4 Stiff integration methods
2. **Advanced numerical linear algebra**
   2.1 BLAS and LAPACK
   2.2 Krylov methods, preconditioning
   2.3 Fast Fourier Transform, block cyclic reduction
   2.4 Spectral and pseudospectral methods
   2.5 Domain decomposition

Syllabus (part 2)

3. **Finite element methods**
   3.1 Sobolev spaces, variational formulation
   3.2 Ritz–Galerkin method, error analysis
   3.3 Computational considerations
4. **Finite volume methods**
   4.1 Riemann problem, Godunov method
   4.2 Lax–Wendroff method, flux limiters, TVD methods
5. **Special topics in scientific computing**
   5.1 Projection method for fluid mechanics
   5.2 Level set and fast marching methods
   5.3 Barnes–Hut algorithm (time permitting)

Note on the schedule

There will be no lecture on March 15 since I am giving a virtual talk at the APS March Meeting.

There will be no lectures on March 1 and March 31, since they are both university wellness days.

The lectures will finish early on Apr 26 to give time for the final project.

Assessment

- 65% – Five homework assignments with equal weighting
- 35% – Final project

Assessment: homework

The homework assignments will involve a combination of mathematical theory and significant programming.

Homework will be due roughly every three weeks – submit a written report and source code via the Harvard Canvas page (linked from main website). The written report should be individually submitted as either a PDF or Word document.

Late homework will be evaluated on a case-by-case basis.

Deadlines are Feb 11, Mar 4, Mar 25, Apr 15, Apr 30.

Writing style and LaTeX

Assignments will be written in LaTeX, which is an excellent platform for writing scientific documents and equations. While completely optional, we encourage you to try and use LaTeX for your assignments. The teaching staff are happy to talk further.

LaTeX is free and available on all major computing platforms. Using LaTeX involves writing a file in a simple markup language, which is then compiled into a PDF or PostScript file. See the excellent *Not So Short Introduction to LaTeX* 2_ϵ[6] for more information.

The original LaTeX files for the homework assignments will be posted to the website for reference.

[6] https://tobi.oetiker.ch/lshort/lshort.pdf

Assessment: code for homework

Code should be written clearly and commented thoroughly.

In-class examples will try to adhere to this standard.

We should be able to easily run your code and reproduce your figures.

Homework assignments: collaboration policy

Discussion and the exchange of ideas are essential to doing academic work. For assignments in this course, you are encouraged to consult with your classmates as you work on problem sets. However, after discussions with peers, make sure that you can work through the problem yourself and ensure that any answers you submit for evaluation are the result of your own efforts.

In addition, you must cite any books, articles, websites, lectures, *etc.* that have helped you with your work using appropriate citation practices. Similarly, you must list the names of students with whom you have collaborated on problem sets. Using homework solutions from previous years is forbidden.

Assessment: final project

The goal of this course is to get you to be a responsible, productive user of numerical algorithms for real-world applications

The best way to demonstrate this is in your final project, worth 35%, to be completed in a group of two or three students[7]

- Use concepts/methods related to the course to solve a problem of interest to your group
- Project proposal in the form of an oral meeting with the teaching staff due on April 2
- Project due at the end of the semester
- Submit a report and associated code

[7] Single person projects may be allowed with instructor permission. Projects with $n \geq 4$ students will be allowed with instructor permission and a statement of how work will be divided.

Textbooks

A variety of the course material will be based off the following books:

- W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery. *Numerical Recipes: The Art of Scientific Computing.* Cambridge University Press, 2007.
- J. Demmel. *Applied Numerical Linear Algebra*. SIAM, 1997.
- D. Braess. *Finite elements: Theory, fast solvers, and applications in solid mechanics*. Cambridge University Press, 2007.
- C. Johnson. *Numerical Solution of Partial Differential Equations by the Finite Element Method.* Dover, 2009.
- T. J. R. Hughes. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis.* Dover, 2000.
- E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer, 1993.
- E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II: Stiff and Differential–Algebraic Problems*. Springer, 1996.

Textbooks

- R. J. LeVeque, *Finite Volume Methods for Hyperbolic Problems*. Cambridge, 2002.
- L. N. Trefethen and D. Bau. *Numerical Linear Algebra*. SIAM, 1997.
- L. N. Trefethen, *Approximation Theory and Approximation Practice*. SIAM, 2013.
- J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer, 2006.
- T. Belytschko, W. K. Liu, B. Moran, and K. I. Elkhodary, *Nonlinear Finite Elements for Continua and Structures*, Wiley, 2014.
- J. S. Hesthaven and T. Warburton, *Nodel Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications*, Springer, 2008.
- B. Fornberg, *A Practical Guide to Pseudospectral Methods*, Cambridge University Press, 1998.

Notes on associated book chapters will be posted to the website.

Poll: C++ familiarity

1. I'm familiar with programming in other languages but I've never touched C++ before
2. I've dabbled in C++, and I could probably write some for-loops and a "Hello world" program
3. I'm pretty confident with C++ and I know about classes and pointers
4. I've used C++ extensively, and I know about advanced features like virtual functions, templating, *etc.*

Poll: Parallel programming

1. I've never done any parallel programming before
2. I've never used OpenMP, but I have done other types of parallel programming before
3. I've heard of OpenMP and I know some basic ideas
4. I've used OpenMP extensively

Poll: Version control

1. I've never used version control
2. I've used some version control software but not Git
3. I've used Git occasionally
4. I've used Git extensively

Poll: Computer access

1. I have access to a quad-core laptop or better.
2. I don't have access to a quad-core laptop.