

MAT 420, Scientific Computing (Fall 2007)

Professor: Eric Kostelich, office PSA 833, telephone (480) 965-5006, email kostelich@asu.edu. Course homepage: <http://math.asu.edu/~eric/mat420>.

Office hours: MW 1:40–2:30 p.m. and by appointment.

Intended audience: Advanced undergraduates and beginning graduate students who are interested in high-performance scientific computing.

Prerequisites: CSE 205, Concepts of Computer Science and Data Structures (or comparable second-semester programming course); MAT 242 or 342, Linear Algebra (or equivalent); MAT 274, Ordinary Differential Equations (or equivalent) is recommended.

Time commitment: *This course will be time-consuming.* Writing and debugging software and learning the material necessary to be productive in a scientific computing environment takes considerable effort. I estimate that this course will require *at least* 9 to 12 hours per week of your time, counting the lectures, labs, and homework.

Texts: There are three suggested textbooks.

1. Mark Lutz and David Ascher, *Learning Python*, second edition (O'Reilly Media, 2004).
2. I. Chivers and J. Sleightholme, *Introducing Fortran 95* (Springer-Verlag, 2000).
3. A. Koenig and B. E. Moo, *Accelerated C++* (Addison-Wesley, 2000).

Other references: The following books are definitive references. I recommend them if you are planning substantial work in scientific programming.

- B. Stroustrup, *The C++ Programming Language*, third (or special) edition (Addison-Wesley, 1997).
- J. C. Adams *et al.*, *Fortran 95 Handbook: Complete ISO/ANSI Reference* (MIT Press, 1997).
- P. J. Plauger, *The Standard C Library* (Prentice Hall, 1991).
- A. V. Aho, B. W. Kernighan, and P. J. Weinberger, *The AWK Programming Language* (Addison-Wesley, 1988).
- R. L. Schwartz and T. Phoenix, *Learning Perl*, 3rd (or latest) ed. (O'Reilly & Associates, 2001);

Also useful:

- E. Quigley, *Unix Shells by Example*, 3rd (or latest) ed. (Prentice-Hall, 2001).
- J. Tisdall, *Beginning Perl for Bioinformatics* (O'Reilly and Associates, 2001).

Software and course content: This course is intended to help you become proficient in developing software for scientific applications in a Posix/Unix environment. We will use the machines in the computer lab in ECA 221 for in-class exercises on Tuesdays. You may log into the machines at other times to complete your projects.

Course meetings: Unless noted otherwise in lecture, we will meet on Thursdays in PSA 309 and on Tuesdays in ECA 221. Attendance at all class meetings in their entirety is expected.

Coverage: The course will provide introductions to the following topics:

- Basic Unix commands: ls, mv, cp, etc., and the Unix filesystem.
- Korn shell programming: environment variables, looping, background jobs, filename globbing.
- Fortran 95, with emphasis on numerical programming, including applications to ordinary differential equations and numerical linear algebra.
- Floating-point arithmetic, including the IEEE-754 standard.
- The Netlib library and GAMS hierarchy.
- The Python programming language.
- C++ programming, with emphasis on template programming.

Comments on the prerequisites: The expected mathematical background includes calculus, linear algebra, and differential equations sufficient to understand what is meant by a solution of a linear system, a singular linear system, Euler's method and the Runge-Kutta methods. The material in the first couple of chapters in any introductory book on differential equations, such as Boyce and DiPrima, will suffice for our purposes. Schaum's Outlines on differential equations and linear algebra are other inexpensive references that you may find helpful.

If you have not taken a course in differential equations, then you will have an opportunity to replace an assignment on differential equations with one on Perl (a widely-used scripting language in bioinformatics, among others).

The expected programming background is a minimum of a one-year sequence in some procedural programming language, such as C, C++, Java, Fortran, Modula, etc. Some familiarity with C/C++ is helpful but not required. We will cover the basics of Fortran, C and C++ in class, but you will need to be familiar with basic concepts like subroutines, looping, dummy arguments, etc.

We will cover the basics of the Unix operating system and shells in class. Some familiarity with Unix or the command-line interface of DOS/Windows operating systems is helpful but not required.

Make-up exams: Make-up exams will NOT be given. Permission to take an exam at a time other than the scheduled one will be granted only at the instructor's discretion. Arrangements must be made *before* the date of the test. Unexcused absences from exams will result in a grade of zero.

Homework and projects: Homework will consist of programming exercises, some of which will be done in class and some of which will be assigned as projects. Information on how assignments are to be turned in will be given in class. Late assignments will not be accepted unless prior arrangements are made with the instructor. Portions of computer lab assignments will be due by the end of class each day. Answers to selected programming exercises will be available from my homepage.

Collaboration policy: You are permitted to work with others on the homework and projects, *provided that you acknowledge your collaborators*. Copying another's solutions or programs without acknowledgment is considered plagiarism and will result in a score of zero for the assignment. If you do most of an assignment yourself but seek help from another student on individual problems, then you must acknowledge assistance on *each* such problem. You may also work with up to two other people on an *entire* assignment; in this case, you will turn in *one* paper and/or program with each co-worker's name on it, and each of you receives the same grade for the assignment. You may choose to work alone on one assignment and with others on another assignment. No collaboration is permitted on the midterm and final examinations.

Grading policy: Grades will be based on a combination of programming exercises (30%), two midterm exams (15% each), a final project (15%), and a comprehensive final examination (25%).

Exam dates: Exams will be held in the computer lab, ECA 221.

Midterm #1:	Tuesday, Sept. 25
Midterm #2:	Tuesday, Oct. 29
Comprehensive final exam:	Thursday, Dec. 6, 12:20–2:10 p.m.

Mathematics Department final exam policy: The Department of Mathematics follows Arizona Board of Regents policy, which states that all final examinations shall be administered at their officially scheduled times. A final exam schedule appears in the Fall Bulletin of classes and online at www.asu.edu/registrar/registration/finals.html. Requests to take the final examination at a time other than the published time will *not* be granted except in emergencies or for reasons of religious practice. The Department of Mathematics reserves the right to require written documentation to substantiate any claim of hardship. Nonrefundable plane tickets, weddings, work schedules, and the like are *not* acceptable reasons for rescheduling final examinations. Please keep this policy in mind when making end-of-semester plans.

Copyright notice: Unless otherwise noted, all instructor-prepared materials, including lectures, handouts, and homework assignments and solutions, are subject to copyright and may not be recorded, reproduced, or distributed without written permission.

Computer lab availability: The mathematics computer laboratory in ECA 221 will be open the following hours:

Monday–Thursday	6:30 p.m.–10:00 p.m.
Saturday–Sunday	2:00 p.m.–6:00 p.m.