This is a get-your-hands-dirty course in how to apply computers to problems in science and engineering. Although we shall treat topics in numerical analysis and programming, as well as in physics, biology, and engineering, the aim is rather to give students the tools and confidence to write big, grungy programs for research or employment.

Dr. David A. Rabson
Physics and Mathematics Building 304
Telephone: 974-1207
Telephone facsimile: 974-5813
e-mail: davidra@ewald.cas.usf.edu
Web: http://ewald.cas.usf.edu

MEETING TIMES: Mondays, Wednesdays, and Fridays 9:00—9:50 (A.M.) in the Physics Building, room 209 (second floor, west). Official OFFICE HOURS to be announced; I am also usually available during the day, evenings, and weekends. OFFICIAL NOTICES IN ACCORDANCE WITH COLLEGE POLICY: you may give away notes for free but may not sell them or tapes. Students who will miss a class meeting due to a planned religious observance are requested to inform the instructor, in writing, by the second class meeting. Letter grades may include + and −.

Not a course in programming

The pre-requisite for this course is CGS 5420, Introduction to Unix and C, or equivalent preparation. This is not a programming or computer-science course; you are expected already to have some facility with the C Programming Language (or to have mastered another computer language and be ready to pick up C on your own, quickly). The first three weeks on data analysis will also review C programming with an aim toward writing readable, portable, and maintainable code. Those unsure of their programming skills should talk to the instructor or count on spending much more time in these first few weeks than for a usual course. Programming is a laboratory skill that takes time to acquire. Computers are perverse, so programming is frustrating. Be prepared.

You are welcome to program in C++ or Fortran-90 if you prefer. While some of the projects could arguably be accomplished efficiently using a package such as Maple or Octave, I shall ask you to use a compiled language instead, as we are going to build up to projects that most definitely cannot be done
with any existing interpreted package. Maple, Octave, and shell scripts are appropriate for auxiliary tasks and plotting; I try to point out where you may use them.

Computing facilities

The Physics Department owns a two-processor Dell computer called physics.cas.usf.edu, managed by CAS (College of Arts and Sciences) Computing Services. You are welcome to develop code on your home or laboratory computers, but you will need to copy your assignments to physics and make sure they compile and work there. The computer runs Linux, a variety of Unix. In my experience, essentially all serious computational physics is done under the Unix operating system, not Windows (you are free to disagree). For security reasons, CAS Computing has installed an ssh daemon on physics; you will need an ssh client and preferably an X-Windows server to log in. CAS Computing will provide information on installing these things on your home computer.

For the final course project, students will use the two “Beowulf-class” computing clusters, Mimir and Wyrd, operated by USF’s office for Research-Oriented Computer Services (http://rocs.acomp.usf.edu). These massively parallel computers (currently comprising a total of 30 nodes, but soon to be expanded) appear to users much like the Cray T3E and IBM SP3 supercomputers run through the National Partnership for Advanced Computing Infrastructure.

Printing

There are vast amounts of documentation you might print, but unfortunately the department can’t afford to let everyone print hundreds of pages on the printer in room 102. I think if each student limits him- or herself to 20 pages a week on the honor system, we should be all right. Feel free to print at home as much as you wish, and if your advisor allows, you can print in your laboratory.

Notes and Texts

Notes for the course are available on-line in the directory /home/5156/packet; all of the required notes and documents will also be made available for purchase in three packets at Pro Copy. It will probably cost you less to buy the packets than to print them out at home. You are not required to print anything, as all the notes can be viewed on-line. However, I strongly recommend that you get at least Packet I, which consists of course notes and assignments. Packet II is intended for people who do not already know Unix, and includes a 26-page operating-system manual, a tutorial for the vi editor, and references for awk and gdb. Additional copyright-protected (but freely-available) documentation is in the directory /usr/local/doc/7ed. Packet III is the reference manual for the CVODE package, which we shall use in the classical-mechanics project.

The course does not follow any book, but I found the following three references useful, all available on reserve and at the Bookstore (prices are from amazon.com). If you have not completely mastered every aspect of C, consider Kernighan and Ritchie “required.” Also students tell me that you should consider Numerical Recipes required.

Press, Vetterling, Teukolsky, Flannery, Numerical Recipes in C, Cambridge, 1992, ISBN 0521575885, approx. $48. This uniquely encyclopedic reference provides a good starting point for almost any numerical problem; I have found myself turning to it for nearly every computational project in my research. Do not use their code. The authors write very unidiomatic C, and the subroutines are known to contain bugs. Use the book to learn the background and to understand the algorithms. Almost every program in the book has a better implementation in the public domain. If you don’t wish to purchase it, the entire book is available free on-line, courtesy of the authors and publisher, at http://www.nr.com.

The following additional references are on reserve at the library:

Thijssen, Computational Physics, Cambridge, 1999, ISBN 0521431085, approx. $58. This book covers several crucial areas of current research not addressed in others textbooks. The appendix provides a quick overview of several useful numerical methods, and there is some overlap with this course in chapters 8 and 14.

Landau and Páez, Computational Physics, Wiley, 1997. This strikes a more even balance between numerical analysis and applications but doesn’t have much overlap with the projects we’ve chosen for this semester.

Giordano, *Computational Physics*, Prentice-Hall, 1997. This book, very similar to Garcia, has also been used in a Matlab-based undergraduate course.

Several people have requested a longer bibliography; here are other references that I or my colleagues have found useful.


Golub and van Loan, *Matrix Computations*, Johns Hopkins, 1996. This is the standard reference for algorithms in linear algebra and the only computer-science book on my shelf.


Wilmott, Dewynne, Howison, *Option Pricing: mathematical models and computation*, Oxford Financial Press, 1993. This was the main reference for the finance project the year we included it; it gives a very practical introduction to free-boundary partial differential equations and numerical techniques for solving them.

**Tentative Course Outline**

week  
1. Histograms and review of C pointers and structures  
2. Probability Distributions, Kernel Smoothing, and (time permitting) Digital Filters in the Time Domain  
3. Working in the frequency domain (fast Fourier transform)  
4. **Project I**: spike sorting in neurobiology  
5. continue spike sorting, wavelet transforms  
6. **Project II**: integrating classical dynamics (ordinary differential equations)  
7. continue classical dynamics, chaos  
8. continue classical dynamics, chaos  
9. **Project III**: quantum magnetism; exact diagonalization of a Heisenberg spin chain  
10. continue magnetism; applications of group theory  
11. continue magnetism; Haldane’s conjecture  
12. **Group Project (IV)**: molecular-dynamics simulation of crack propagation on a parallel computer  
13. continue group project  
14. continue group project  
15. continue group project

Two years ago, we completed two additional projects on experimental data acquisition and control using Labview and on mathematical finance. The Labview project has been moved to the course on Electronics and Instrumentation. The mathematical-finance project concerns the pricing of American put and call options on the stock exchange; notes are included in Packet I, and students are welcome to work on the project for extra credit. The more leisurely pace afforded by fewer projects should help students get more out of each. I expect that a somewhat different set of projects will be treated each year.

**Assignments**

Grades will be based on exercises and projects assigned; there are no examinations. Unless otherwise indicated, assignments will be due by that midnight that falls between Sunday and Monday the day before
we move on to a new topic (see course outline). The clock on physics.cas.usf.edu will be considered official. To earn an A, your programs and reports should be complete and correct, including handling important special cases, well-documented in grammatical English, easily understood, and concise.

Cooperation and Collaboration

Students are encouraged to discuss their problems, results, and methods and to assist one another in finding bugs. Indeed, for some projects I will ask you to compare your results, so cooperation is required. Except for the final group project, however, such cooperation must not include plagiarism or the exchange of source code, either by electronic mail or in printed form: each student must write her own programs and her own reports.

The final group project is different. For this project, the whole class will earn a single grade (although any student who does not pull his or her weight will be marked down). After I have explained the problem and offered some suggestions, the class will have to decide for itself how to split the problem into manageable pieces, assigning each member certain tasks. Presumably, some students will accept additional responsibility for coordinating the project. Editing will also be important: at the end, I expect a well-written report that reads continuously, not like a compendium of ten individual sections.