Astronomy 615 - Spring 2014
"Computational Astrophysics"

Instructor

Prof. Massimo Ricotti
Office: PSC 1156
E-mail: ricotti@astro.umd.edu
Phone: (301) 405 5097
Office hours: TBD or by appointment
Class web page: http://www.astro.umd.edu/∼ricotti/NEWWEB/teaching/ASTR615-14.html

Class Schedule

Lectures on Tuesday and Thursday from 12:30pm to 1:45pm
Room CSS 0201

SYLLABUS

Preface

This course does not require previous programming experience but if you never learned a compiled programming language, by the end of this course, you will.

I discourage the use of any programming language other than Fortran77, Fortran90, C, C++. The use of Java or python can be discussed. Interpreted programming languages like Matlab, Matematica and IDL are not allowed in this course. The reason is twofold:

i) The term "computational astrophysics" typically refers to running cutting-edge, computationally intensive applications that require high-level efficiency and parallelization. These custom built codes need to run on supercomputers and are typically written in C or Fortran and are parallelized using OpenMP or MPI. Recently, has also become possible to run supercomputer codes on GPU using CUDA or OpenCL languages (we will touch on that).

ii) The course focuses on understanding the mathematical methods involved in writing efficient numerical algorithms. Your goal is to write these algorithms from scratch as a learning process. However, several interpreted programming languages have these algorithms already built in: they work like a "black boxes" with an input and an output, thus are not suited for teaching the numerical methods hidden inside the "black box".

My "mother tongue" is Fortran, but I can also speak C as a second language, and if necessary I can understand some Java or python. Solutions to the homework will be in C.

The level of the class and the topics covered will partially depend on the student's previous familiarity with low-level programming languages. Along the way I may complement the lectures with power point presentations on topics that the class is especially interested in. I will keep the webpage updated and link all the course material there. Laptop computers are allowed in class during the first part of the course.

Course Description

This course will provide the astronomy student with a basic knowledge of numerical methods in astrophysics. By the end of the course students should be comfortable working in a Unix environment, compiling and running codes, and employing a variety of visualization techniques to analyze the results. This process will be motivated by concrete examples of modern problems in astrophysics that demand numerical approaches.

As mention above, the material covered will depend on the existing level of computer sophistication among the class participants. However, in broad outline the major course topics will include linear algebra, root finding, least-square fitting, Monte Carlo methods, numerical integration, N-body methods, fluid dynamics, FFTs and time-series analysis.

Recommended Texts

There is no required text for this course. The following recommendations may be helpful to you. Note that much of the course material will follow, *Numerical Recipes*, which is available online. Most in-class programming examples will be in C, but you are free to chose from any suitable languages for completing the assignments.

Aarseth, S. J. 2003, "Gravitational N-body Simulations: Tools and Algorithms," Cambridge Univ. Press.

Hockney, R. W., and J. W. Eastwood 1988, "Computer Simulation Using Particles," Hilger. [Out of print?]

Kernigan, B. W., and D. M. Ritchie 1988, "The C Programming Language" (2nd ed.), Prentice-Hall.

Peek, J. D., et al. 1997, "Learning the Unix Operating System (Nutshell Handbook)" (4th ed.), O'Reilly.

Prata, S. 1998, "The Waite Group's C Primer Plus" (3rd ed.), Howard W. Sams & Co.

Press, W.H. et al. 1992, "Numerical Recipes in Fortran [or C or C++]" (2nd ed.), Cambridge Univ Press - visit the website at *http://www.nr.com/*.

Yee, H.C. 1989, "A class of High-resolution Explicit and Implicit Shock-Capturing Methods", Tech. Report Lecture Series 1989-04, von Karman Institute of Fluid Dynamics [difficult to find?]

Course Grading

A+	$> 96\%$
A	from 96% to 92%
A-	from 92% to 88%
B+	from 88% to 84%
B	from 84% to 80%
B-	from 80% to 76%
C+	from 76% to 72%
C	from 72% to 68%
C-	from 68% to 64%
D+	from 64% to 60%
D	from 60% to 56%
D-	from 56% to 52%
F	below 52%

Grades will be determined by homework assignments, one final and midterm exams. The homework assignments are at the core of this class, and are worth 50% of your final grade.

Homework	50%
Midterm	20%
Final	30%

There will be no extra points assignment or curve on the final grades. Some adjustment to scores depending on the class average may be necessary; however, any adjustment will be to lower the percentages given above, never to raise them.

Assignments

Most assignments involve programming exercises. To make evaluating your work easier, you must e-mail me a single "stand-alone" file containing all your work by the start of class on the day the assignment is due. The file (e.g., a gzip tar archive or a zip file) when uncompressed must produce a folder (named by your last name), containing a suitable formatted response (PDF is best, as Word has compatibility issues especially with embedded figures) to the questions posed in the assignment, along with a description of the remaining contents of the file, including, as needed, instructions on compiling and running any source code. Ideally a Makefile should be provided. Any static graphical output (plots, etc.) should be embedded in the response document.

I will compile and run your code with a set of test parameters to ensure correct functionality and error handling. I will also consider your coding style when evaluating your work. Assignments that are late will automatically incur a 20% penalty unless there are extenuating circumstances. Late assignments must be completed before the solutions are posted on the web to get any credit.

You may work in groups to discuss programming strategy, but you must submit your own solution to each assignment. Note that, just as for written prose, it is necessary to cite the source of any algorithms you use in completing assignments. This includes Numerical Recipes routines that you use.

Course Policies

The Honor Code

The University of Maryland, College Park has a nationally recognized Code of Academic Integrity, administered by the Student Honor Council. This Code sets standards for academic integrity at Maryland for all undergraduate and graduate students. As a student you are responsible for upholding these standards for this course. It is very important for you to be aware of the consequences of cheating, fabrication, facilitation, and plagiarism. For more information on the Code of Academic Integrity or the Student Honor Council, please visit http://www.shc.umd.edu/. To further exhibit your commitment to academic integrity, remember to sign the Honor Pledge on all examinations and assignments: "I pledge on my honor that I have not given or received any unauthorized assistance on this examination (assignment)." (NOTE: for the assignments in this course, you may include this statement in your e-mail to me, or the enclosed writeup.)

Course Evaluation

Your participation in the evaluation of courses through CourseEvalUM is a responsibility you hold as a student member of our academic community. Your feedback is confidential and important to the improvement of teaching and learning at the University as well as to the tenure and promotion process. CourseEvalUM will be open for you to complete your evaluations for fall semester courses between Tuesday, April 29 and Friday, May 14. Please go directly to the website (http://www.courseevalum.umd.edu/) to complete your evaluations starting April 29. By completing all of your evaluations each semester, you will have the privilege of accessing online, at Testudo, the evaluation reports for the thousands of courses for which 70% or more students submitted their evaluations.

Students with Special Needs

Students with a documented disability who wish to discuss academic accommodations should contact me as soon as possible.

Tentative Course Outline

Date	Lecture	Reading	(NRiC)
#1	Jan 28	Introduction to the course and survey	–
#2	Jan 30	Computer architecture	–
#3	Feb 04	Introduction to UNIX	tutorial
#4	Feb 06	Introduction to C	1.1-1.2, tutorial
#5	Feb 11	Examples in C and debugging	1.1-1.2, tutorial
#6	Feb 13	Introduction to visualization	tutorial
#7	Feb 18	Data representation	1.3
#8	Feb 20	Linear algebra, part 1 (Gauss-Jordan elimination)	2.0-2.3
#9	Feb 25	Linear algebra, part 2 (LU & SVD decomposition)	2.4-2.6
#10	Feb 27	Root finding in 1-D	9.0-9.1, 9.4, 9.6
#11	Mar 04	Root finding in multi-D, and numerical differentiation	5.7
#12	Mar 06	Statistics and the K-S test	14.0-14.3
#13	Mar 11	Least-squares fitting	15.0-15.2, 15.4-15.5
–	Mar 13	MIDTERM	–
–	Mar 18	Spring Break	–
–	Mar 20	Spring Break	–
#14	Mar 25	Random numbers and cryptography	7.0-7.2
#15	Mar 27	Numerical integration	7.6, 4.0-4.4, 4.6
#16	Apr 01	Integration of ODEs, part 1 (IVPs)	16.0-16.1
#17	Apr 03	Integration of ODEs, part 2 (leapfrog)	–
#18	Apr 08	Integration of ODEs, part 3 (stiff ODEs & 2-pt BVPs)	16.6, 17.0
#19	Apr 10	Integration of ODEs, part 4	–
#20	Apr 15	N-body techniques, part 1	–
#21	Apr 17	N-body techniques, part 2 (PP)	–
#22	Apr 22	N-body techniques, part 3 (PM)	19.0, 19.4-19.6
#23	Apr 24	N-body techniques, part 4 (tree)	–
#24	Apr 29	Integration of PDEs, part 1 (ell & hyp)	19.0-19.1
#25	May 01	Integration of PDEs, part 2 (hyp & par)	19.2
#26	May 06	Fluid dynamics, part 1 (eqns)	–
#27	May 08	Fluid dynamics, part 2 (methods)	19.3
#28	May 13	Parallel Computing (CPU and GPU)	–

Final Exam: TBD. Due to travel commitments my preference is May 15th or 16th in CSS 0201

Note: check periodically online for up to date Course outline.