Course Syllabus - Special Projects in Astronomy

Introduction to Astronomical Programming

ASTR 288P
Spring 2018

Learning Outcomes

The goal is for the student to be comfortable using the UNIX environment and introduce students to scientific programming in Python and C (with emphasis on Python). Students will learn how to operate within an UNIX environment and run commands within a Linux, Mac, or Windows Bash terminal (lab machines run Scientific Linux 6.7). Students will learn the basics of scientific programming including, code correctness and good programming style, numerical methods/analysis, and visualizing results. Students will also be made comfortable installing and using open source software, which includes collaborating and sharing their own code. Exact details of the material covered will depend on existing level of experience of the class participants.

A more advanced version of this class is ASTR415 (Computational Astrophysics).

After successfully completing this course you will be able to:

- Work within a Unix and terminal environment.
- Work with basic scripts and python coding.
- Install open source software.
- Visualize scientific data.
- Analyze scientific data.

Books and Other Resources

Course website: http://www.astro.umd.edu/~krmurphy1/ASTR288/
Course GIT repository: https://github.com/kylermurphy/astr288p_2018
Books (recommended):
Numerical Recipes (used in ASTR415) http://numerical.recipes/
References:
See the course website for a list of references regarding Unix/Linux, Python, C, and Windows Subsystem for Linux

Dr. Kyle R. Murphy
krmurphy@umd.edu
ATL 0251A

Class Meets
Mondays
4:00 pm – 5:50 pm
ATL 0224

Office Hours
Mon 3:00-4:00 pm
and by appointment

Prerequisites
Permission of CMNS-Astronomy department required. Must have completed or be concurrently enrolled in ASTR121

Course Communication
All communication regarding the course will be done through the email (as provided by each student).

Helpful guidance on writing professional emails (ter.ps/email).

Campus Policies

It is our shared responsibility to know and abide by the University of Maryland's policies that relate to all courses, which include the following topics:

- Academic integrity
- Student and instructor conduct
- Accessibility and accommodations
- Attendance and excused absences
- Grades and appeals
- Copyright and intellectual property

Please visit www.ugst.umd.edu/courserelatedpolicies.html for the Office of Undergraduate Studies' full list of campus-wide policies and follow up with me if you have questions.

Activities, Learning Assessments, & Expectations for Students

This class is meant to introduce students to the basics of Scientific and Astronomical Programming. Students will be guided in their learning with lectures, in-class problems, and formal assignments. Lectures will occur in most classes although time will also be taken to work through in class exercises and programming assignments. Class participation is expected during the in-class exercises.

Students can expect between 4-6 homework assignments. These will be assigned roughly every 2 weeks with approximately 2 weeks to complete the assignment. Completed assignments are due before each class through submission on URSA.

Students will have access to the computer lab (ATL 0224) with an account on the Linux system, however students are also able to use their own laptops in class and for homework assignments. Access to the lab will be available when it is not otherwise in use.

Grades

Grades are not given but earned. Your grade is determined by your performance on the learning assessments in the course and is assigned individually (not curved). If earning a particular grade is important to you, please speak with me at the beginning of the semester so that I can offer some helpful suggestions for achieving your goal.

Late work will not be accepted for course credit so please plan to have it submitted well before the scheduled deadline. I am happy to discuss any of your grades with you, and if I have made a mistake I will immediately correct it. Any formal grade disputes must be submitted in writing and within one week of receiving the grade.

Learning Assessments	Category Weight
Participation:	10%
Assignments:	90%

Note: Not all assignments will be weighted equally. As the course progresses assignments will become more complicated and hence will be worth more. Assignments will be graded on a 'points', the cumulative points of all assignments will then be weighted to 90% of the final grade.

Final letter grades are assigned based on the percentage of total assessment points earned. To be fair to everyone I have to establish clear standards and apply them consistently, so please understand that being close to a cutoff is not the same thing as making the cut ($89.99 \neq 90.00$). It would be unethical to make exceptions for some and not others.

Final Grade Cutoffs									
+	97.00%	+	87.00%	+	77.00%	+	67.00%		
A	94.00%	B	84.00%	C	74.00%	D	64.00%	F	<60.0%
-	90.00%	-	80.00%	-	70.00%	-	60.00%		

Course Schedule

A tentative schedule of the course schedule is provided below. As the course progresses and we gauge the speed of the course the schedule will be fleshed out and new topics will be added.

TOPIC			
Week 1		Jan 21 – Jan 27	No Class
Week 2	Lecture 1	Jan 28 – Feb 3	Introduction and Assessments
Week 3	Lecture 2	Feb 4 – Feb 10	Unix (shells, files and directories), GIT
Week 4	Lecture 3	Feb 11 – Feb 17	More on Unix (file commands, scripting, permissions)
Week 5	Lecture 4	Feb 18 – Feb 24	Unix/Python (remote access, paths, variables, miniconda)
Week 6	Lecture 5	Feb 25 – Mar 3	Python (jupyter, variable types, control flow)
Week 7	Lecture 6	Mar 4 – Mar 10	Python (arrays, plotting) **Guest Lecture**
Week 8	Lecture 7	Mar 11 – Mar 17	Python scientific programming and numerical methods
Week 9	Lecture 8	Mar 18 – Mar 24	**Spring Break, no lecture**
Week 10	Lecture 9	Mar 25 – Mar 31	Python scientific programming and numerical methods
Week 11	Lecture 10	Apr 1 – Apr 7	Python scientific programming and numerical methods
Week 12	Lecture 11	Apr 8 – Apr 14	Python scientific programming and numerical methods
Week 13	Lecture 12	Apr 15 – Apr 21	Python scientific programming and numerical methods
Week 14	Lecture 13	Apr 22 – Apr 28	Python scientific programming and numerical methods
Week 15	Lecture 14	Apr 29 – May 5	Final Presentations
Week 16	Lecture 15	May 6 – May 12	Last class, recapping (continued presentations if required).

Topics to be covered in scientific program and numerical recipes include (but isn't necessarily limited to):

- root finding
- curve fitting
- numerical integration
- numerical differentiation
- time-series analysis
- image processing, plotting and multidimensional arrays