Course Website: http://brahms.phys.westminster.edu/comp

Course Meetings: MW 3:10–4:40PM, Hoyt 113
Instructor: Prof. Rob Knop
Email: knopra@westminster.edu
Office: Hoyt 121, x7201
Office Hours: MWF 10:30-11:20AM, by appointment, and when you can find me

Course Overview

In the last few decades, computers have become an essential tool of physics. What’s more, the nature of physics research is such that frequently, there is no off-the-shelf software that will just do exactly what you want. You must either write your own code, or be able to adapt and use code that is close to what you want.

Many problems in physics cannot be solved in closed form. Only for a few problems can you derive an algebraic expression for the evolution of the system as a function of time. Other problems require “numerical” solutions, where the problem is solved by a large number of small calculations, be it the evolution of the system through small time steps, successive approximations, or other techniques involving a very large number of computations. Performing a very large number of repeated computations is, of course, exactly what computers are best at. A fully study of physics in the modern age is not complete without some understanding of how numerical techniques are used in the analysis of physics problems. This is why we have integrated a computational component into the introductory physics class here, and this is why you are taking this class.

The goal of this course, together with the second semester (Physics 332), is for you to learn how to effectively use computers in the analysis of physics problems. It will focus on writing your own code, including using scientific programming libraries, for these problems. As such, part of the goal of this course is for you to become a better programmer, and some of the topics in the course will be focused more on the parts of programming that are most important for physics purposes. We will both be looking at numerical algorithms used for the theoretical analysis of physics problems, and at the use of numerical methods in analyzing data.

Course Outcomes

At the end of the semester, the successful Physics 331 student will be able to:

- Write code in Python, including code that makes use of functions and classes.
- Produce two- and three-dimensional visualizations of physics data and the results of physics computations.
- Numerically solve a series of linked ordinary differential equations describing the evolution of a physical system over time.
• Handle and process computational data structures that describe vectors and matrices.

• Fit an empirical model to a set of data by writing code that harnesses numerical libraries implementing nonlinear minimization methods.

• Evaluate a numerical method to determine how precise its calculations are.

• Debug code in order to produce a functioning program.

• Effectively use computational techniques in other more advanced physics classes, and as a focus for or a supplement to their capstone research.

Students will learn this material through lectures and demonstrations in class, and also through their own experimentation and exploration of documentation available for Python on the web. Their performance will be assessed through evaluation of homework assignments involving computer code they will write.

Computers and Software

We will be using Python for most things in the class. Python is a well-supported and popular open-source programming language that is (relatively) easy to use, and yet extremely powerful. It is available on all modern computer platforms. (Support on phones and tablets exist, but is more limited or experimental.) Additionally, it has a wealth of scientific and numerical libraries available, and is increasingly being adopted in the physics and astronomy communities. Most importantly, it is the same language that is integrated into the introductory physics course here.

"Wait, why can’t I just use Java!" those of you who have taken or are taking Computer Science 151 are crying. “That’s the language we’re learning in our CS class!” In fact, it’s probably better for you to learn two languages at once. What’s important about computer programming is understanding its structure and concepts, not the specific individual syntax of each language. The basic paradigm you’ll be using in both cases—Object-Oriented Programming—is the same. As such, what you learn in each class will be useful for the other class. The most important differences are just the details of individual syntax, which you can always just look up. Myself, I have used enough different programming languages that I’m always forgetting, for example, how you specify the logical AND in the language I happen to be using at the moment. That being said, there are some important structural differences between Python and Java, which I will talk about in class. Also, there are other programming paradigms out there, such as Functional Programming, that are better supported by other languages.

We will also be using a number of additional packages—NumPy, SciPy, and Matplotlib, at least—that are extensions to Python. You can install all of these things (with some effort) on your own computer following the links I’ve given you on the course website. However, everything is already installed on all of the computers in the classroom. You have a key to this classroom, and can use the computers there at any time throughout the semester. What’s more, we’ll be using Linux as our environment, and running a lot of what we do from the Linux command line, throughout the course. You can, of course, install Linux on your own computer, and I’d be happy to help you with this; getting most of the rest of what you need for this course is very easy with most modern Linux distributions, more so than it is on Windows or (especially) MacOS.

In addition to Python, we may be doing a little bit of work with Maxima, the open-source computer algebra system.
Assignments and Grading

In the second semester of this course (Physics 332), the primary focus of the course will be a project you select yourself. This semester, we are focusing more on learning a lot of basic techniques and algorithms (including the algorithms that will be most useful for Physics 351 in the Spring). One fifth of your grade will be based on the effort I observe you put into the class; are you really trying, are you really engaged?. For the other four-fifths, I’m taking a “mastery” approach. For each topic listed below, I will note whether you have achieved “borderline” (C), “adequate” (B), or “excellent” (A) mastery of the topic, based on the work you show me, and a short oral in-person quiz. You will turn in this work when you have completed it, and take the oral quiz when you are ready. If you are not happy with your score on each topic, you may improve it with further work and a retake of the quiz. However, make sure to manage your time! If you find yourself taking a lot of oral quizzes in the last two weeks of the course, you are probably in trouble. Make sure to pace yourself, and get at least “adequate” scores on most of the topics.

The “Progress” part of your course grade will depend on how many topics you master. The total number of points you earn, summed over all topics will be divided by 110, the result giving your grade (0.7–0.8 is in the C range, 0.8–0.9 is in the B range, etc.) for the “Progress” part of your course grade. If you look at the numbers below, you will notice that there are more than 110 points available. As a result, you may choose to view some of the topics as optional, if you wish. For example, if you achieve “borderline” proficiency in all topics, but don’t achieve any proficiency in Advanced NumPy, Chaos, PDE Stability Analysis, or Graphical User Interfaces, you will earn 68 points, giving you a 72% (C-) for the “Progress” part of the course. If you achieve “adequate” proficiency in that same set of topics, you will have 81% (B-) for the “Progress” part of the course. And, yes, if you do very well and go all out, it is possible (although it will be difficult and require a lot of work!) to get more than 100% for the Progress part of your grade.

Listed here are the topics, and the number of points available for that topic, in the order I suggest you address them. (Certainly, you will need to complete the first three topics before any of the others.)

<table>
<thead>
<tr>
<th>Topic</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Python</td>
<td>10</td>
</tr>
<tr>
<td>Intermediate Python</td>
<td>10</td>
</tr>
<tr>
<td>Use of NumPy</td>
<td>10</td>
</tr>
<tr>
<td>Plotting with Matplotlib</td>
<td>10</td>
</tr>
<tr>
<td>Advanced NumPy</td>
<td>5</td>
</tr>
<tr>
<td>Finding Roots</td>
<td>5</td>
</tr>
<tr>
<td>Solving ODEs</td>
<td>20</td>
</tr>
<tr>
<td>Evaluating ODE Solution Quality</td>
<td>10</td>
</tr>
<tr>
<td>Using Scientific Libraries</td>
<td>10</td>
</tr>
<tr>
<td>Chaos</td>
<td>5</td>
</tr>
<tr>
<td>Solving PDEs</td>
<td>10</td>
</tr>
<tr>
<td>PDE Stability Analysis</td>
<td>5</td>
</tr>
<tr>
<td>Data modeling</td>
<td>10</td>
</tr>
<tr>
<td>Graphical User Interfaces</td>
<td>5</td>
</tr>
</tbody>
</table>

I will present a lot of the material for most of the topics in class. I will also post supplementary
reading for many of the topics. On the course website, I will be posting exercises for learning purposes, and assignments to be turned in for evaluation, for each topic. I’ll also give a brief outline of what I’ll expect you to be able to demonstrate in the oral quiz for each topic.

Attendance

Attendance at all class meetings is mandatory. Your active engagement in class, working towards mastering the course material, will graded as part of the “Class Participation” score. Of course, we are all adults; if you are ill, or even if you are too exhausted to make good use of class time, you may make the decision that your time is better used sleeping. Making good use of course time is more important than simply being there. A student who misses two meetings but works very well when she’s there will score higher than the student who always shows up but does not mentally engage. However, if you do miss any meetings of the course, you are still responsible for everything that happened during that course meeting, including any announcements about changes in due dates for assignments. While changes of these sorts of things will generally be announced on the course website, you should speak with a friend after any missed course meeting to make sure that you are up to date with the course. In any event, I will not rehash or summarize what happened in class for you if you miss class.

Academic Integrity

You are expected to comply with Westminster College’s policy on Academic Integrity, as described in the College Catalog. If you are suspected of violating this obligation, then you will be required to participate in the standard procedures for handling academic integrity violations.

In this course, you are expected to write your own code. This does not mean that you cannot look at another students’ code; indeed, working with other students, especially when it comes to helping each other debug your code, can be very illuminating. Likewise, web searches for help with coding techniques can be very useful. However, you may not simply copy code from somebody else—either somebody else on in the class, or somebody whose code is posted on the network. If you turn in code that is copied from another person’s code, claiming that you “wrote it together”, this will be considered plagiarism. While it is possible to collaborate on code, and indeed, some software development houses rely on that technique, while you are still learning how to code you will not learn as much if you aren’t forced to think through things yourself. For that reason, each of you must construct and turn in code that is your own.

Special Accommodations

(Borrowed from a syllabus written by Jamie McMinn.)

If you have special needs that may affect your academic life during this semester, you should speak to a member of Disability Resources at x7192. Accommodations that are consistent with college policy will be considered. If you need additional assistance with your coursework and study habits, then you should contact the Learning Center at x6700. For issues that may affect your personal life this semester, please contact the counseling Center at x7340.