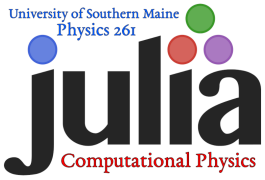
Physics 261: Computational Physics — Syllabus

Paul Nakroshis

Fall 2025



Physics 261: Computational Physics is an introductory course on scientific programming using the Julia programming language. We will work primarily using Jupyter Notebooks and Pluto Notebooks, which allow you to mix text, graphics, mathematical equations (using LaTeX) and, of course Julia code, in your reports. By the end of the semester, you should feel comfortable using Julia to solve everyday problems, from data analysis to numerical simulation. Along the way, you will gain experience with the linux terminal, and logging into a JupyterHub server.

Instructor Contact Information

Paul Nakroshis
Dept of Physics
Office: 224 Science **Lab:** 252 Science
Portland Campus
EMail: pauln at maine dot edu
Web: portlandphysics.me
Github: https://github.com/paulnakroshis
Office Phone: 207-780-4158
Student Hours: by appointment

This document & other resources

Should you lose this syllabus, an electronic .pdf version of this file (with clickable hyperlinks) is available online at the course homepage which can be found at
portlandphysics.me/phy261.

Introduction

Suppose you take an introductory calculus-based physics course; you will have learned all about Newton's laws of motion and solved many problems with an analytical approach. However, one problem you will not have solved is a simple problem of throwing a ball while including air resistance. One of the reasons you didn't solve this problem is that it's impossible to solve analytically, since air resistance is a non-linear force[1] However, adding air resistance which is proportional to the square of the particle's velocity, while impossible to solve analytically, is not so complicated to solve computationally.

[1] Actually, it's even worse—air resistance is not really a simple function of velocity at all; for low velocities, one can approximate the air resistance as linear, and but as the speed increases, it's not even correct to treat it as a simple function of the velocity to a fixed power.

So, one of the great uses of computation is to be able to solve problems via computer that are difficult or impossible to solve with pen and paper. Such problems can range from solving non-linear differential equations, to systems which involve random processes (for instance, diffusion, or disease propagation), and, of course, the everyday work of processing and visualizing data. In this class, I will introduce you to the Plots.jl package (and possibly Makie.jl) (useful for visualizing data and functions), and build up your facility with the Julia language so that you will walk away with a toolset that you can use throughout your scientific career.

Background

"There is a computer disease that anybody who works with computers knows about. It's a very serious disease and it interferes completely with the work. The trouble with computers is that you 'play' with them!"
—Richard P. Feynman

For over a hundred years, physics departments around the world have taught undergraduate physics by emphasizing the solution of problems which admit of an analytical solution. However, in recent decades, computer power has increased significantly and a modern laptop has the speed that far exceeds that of a mainframe computer from decades past. Hence, given that most undergraduates now own a laptop computer, it makes no sense to restrict our teaching to analytically solvable questions. Many physics departments now have a computing requirement, or even their own computational physics course, but few go beyond this.

At USM, we are modifying our curriculum to include computing throughout our major's degree. The first step in this is to teach computational physics every in the freshman year so that entering students have the needed experience to use computation throughout all their upper level courses. Because computing is important to many disciplines—engineering, linguistics, business, mathematics, statistics, biology, chemistry for example—our computational physics course now satisfies the computing requirement for Chemistry, Mathematics, as well as Physics.

Skills/Learning Outcomes

By the end of the semester, my expectation is that you will gain experience in the following:

1. The Bash Shell (basic commands like ls, cd, mv, rmdir, etc)
2. Logging into a JupyterHub server and using the Jupyter Lab and Jupyter Notebook interfaces
3. Starting up and working in a Pluto.jl notebook
4. Learning how to install and maintain a working Julia distribution on your own computer
5. Uploading data and .ipynb files to the server

6. Using Github for version control
7. Learning the basics data types, and control structures in Julia
8. Reading in and writing data files in multiple formats
9. Become proficient in using Markdown syntax and incorporating basic LaTeX code to typeset mathematics.
10. Become proficient in technical written communication.
11. Learn how to use the Euler, Euler-Cromer, Verlet, and Runge-Kutta methods for solving ordinary homogeneous and inhomogeneous differential equations.
12. Using simulation to solve random systems (random walks in 1 and 2 dimensions, percolation theory)
13. Use Julia to read in data, make technical plots, perform fits to the data, as well as the uncertainties in the fit parameters.
14. Simulate the motion of simple systems such as projectile motion, planetary motion (with and without air resistance)
15. Explore the behavior of chaotic systems such as a driven non-linear oscillator, or the logistic map
16. Work together on a team to solve a larger programming project, all coordinated using version control.
17. Walk away from class with a computational toolset that can be used for all further coursework, and provides a framework of understanding relevant to a STEM Career and further scientific work.
18. Leaving class with a significant library of code snippets and notebooks which will be a useful personal resource.

Required Textbooks

Think Julia, by Allen Downey, Ben Lauwens, 2018.
ISBN 978-1492045038
This book is also available for free online at
https://benlauwens.github.io/ThinkJulia.jl/latest/book.html

Required account

I will ask every student to sign up for a free account on GitHub Click here to setup a free account; I suggest a username of firstLast; i.e. albertEinstein if your name happens to be Albert Einstein. There is no rush to create this account; when the time comes, it will be part of in-class work.

Atendance Policy

I expect that everyone will be at every class except in extenuating circumstances. In every class, you will likely be working together on a team, and your presence in class is important. If I find that you are missing class too often (i.e. more than three times), you can expect that I will talk with you and that you will likely receive a lower grade for the course, or asked to leave if this repeated absence is coupled with poor quality work written work.

Outside Help/Student Hours

I have an office in room 224 Science (the "bridge") and a lab in room 252 science—you have to go through room 250 to reach it. In general, if my office or laboratory door is open, I am happy to help you, so feel free to stop in and ask questions. I have set aside several hours where I will make a point to be in my office. Please take advantage of my willingness to help you! I can be much more effective one-on-one than I can in front of the whole class.

Here are some times I am not available:
Tu Th 9-12:15 (taking French 301, and then teaching Quantum Mechanics)
Friday afternoons are often taken up by Faculty Senate or Department Chairs meetings, but typically no more than twice a month.
Other than these times, I am pretty easy to find either in my office or lab.

Grading

I am unfortunately contractually obligated to submit a grade for every student. Grading is subjective, and I do my best to submit a grade which represents my sense of your level of understanding and your level of improvement in understanding for the course.

This course will consist of shorter **assignments** (more toward the beginning of the course) and longer **projects** which will require a formal report. In addition, there may be occasional quizzes, and likely group reports. You will get regular feedback on each assignment and each report, as well as on your final written report. In attempt to make this quantitative, Table 1 shows the grading scheme (Late assignments and project reports **will lose 25% per day late.** Your work in this class is in learning the Julia Language ecosystem and applying your knowledge of this to simulate physical systems. This task is your homework, and is the overarching skillset you will be building outside of class.

Important note on written communication

It is important to note that your grade on your assignments is based on **both** the quality of the code you write **and** the quality of your written explanations. Your assignment solutions should walk the reader through your thinking and

you should think of your audience as a student in a similar course that has never seen the particular problem you are solving and needs detailed and clear explanations of your thinking process. Unless specified otherwise, a report with insufficient written discussion (using the markdown features in Jupyterlab and/or Pluto) will lose a significant amount of credit.

Item	Points
Attendance × Perceived Effort	100
Assignments* & Projects*	700
Quizzes	100
Final Project	50
Final Project presentation	50
TOTAL	1000

Table 1: Grading Scheme for the course. *Late assignments and projects **will loose 25% per day late**.

Items to include in Formal Reports

Some homework will be relatively straightforward exercises; others will be longer projects. On these more formal reports, here are guidelines for your submitted report.

(1) Introduction

The introduction should give an overview of the problem and an indication of where it fits into the subject of physics.

(2) Physics & Numerical Method

Describe the background physics of the problem, and detail the algorithm that is used to solve the problem. This section should also list relevant snippets of your code to show how it is implemented. A full listing of your code should always be attached as the last section of your report.

(3) Verification

Tell what you did to verify that the program gives correct results; this typically involves showing that your code gives reasonable results for simple cases where an analytic solution is known or obvious. Generally speaking there should almost always be more than one test used to verify program integrity. You have to convince the reader that your code reproduces known or sensible results.

(4) Results

Present the results of running the program to demonstrate the behavior of the system under different circumstances. Results might be presented in graphical form or as tables, as appropriate. Be sure that results that are presented are labeled properly, so that the reader can figure out what has been calculated and what is being displayed. Make sure that all figures and tables should have descriptive captions, and all axes have axes labels with units. Your results will include marksdown cells explaining your work as you go. Remember, you are

having a once-sided conversation with the reader, and it behooves *you* to fully paint a clear picture of your thinking process and write this in an engaging manner.

(5) Conclusion

Present a discussion of the physical behavior of the system based on your simulations, remind the reader of what you have shown, and answer any special questions posed in the assignment.

See this link for a different take on writing a report. It's worth a read.

Dates	Topics
03 & 05 Sep	Course Intro, accessing server; setting up laptop Using the REPL, using Jupyterlab, using Pluto.
08 —12 Sep	Introduction to Julia (I), JupyterLab, Linux & LaTeX
15 — 19 Sep	Introduction to Julia (II): variable types, functions
22 — 26 Sep	Introduction to Julia (III): Loops, algorithms
29 Sep — 03 Oct	Introduction to Julia (IV): Plots.jl, Make.jl
06—10* Oct	Pluto Notebooks, github
15 & 17 Oct	Kinematics in 1D: Air Resistance (I)
20 — 24 Oct.	Kinematics in 2D: Gravity (II)
27 — 31 Oct	Kinematics in 2D: Gravity (III)
03 —- 07 Nov	Simple Harmonic Motion (I)
10 —14 Nov	Simple Harmonic Motion (II)
17 — 21 Nov	Data Analysis: Fitting with LsqFit, CurveFit
24 Nov.	Random Processes: diffusion (I) Random Processes: diffusion (continued)
01 — 05 Dec	To Be Decided (time for final projects?)
08 —12 Dec	Final Projects and Presentations

Table 2: Likely schedule of topics; almost everything subject to change!

* On Oct 10, I will not be in class, but you will have group assignments using github and I will hopefully have one of our physics majors help out.