Final report : Application of Python programming for University Physics course

Goal and Strategy:

The goal of this project was to give the freshman and sophomore students majoring in physics, chemistry, mathematics, and computer science a first hand experience of solving problems in physics on computer. I did a survey before the beginning of the semester to understand what was the level of "computer literacy" in the student population. It became apparent that a large fraction of students were co-enrolled in an introductory Python course, however only thirteen students have prior knowledge of computer programming of any sort. Most students were also using some Windows operating system, and a handful of them were using either a Mac or a Chromebook. The XSEDE Grant was a great asset to provide a level playing field for all of them. I launched a Littlest JupyterHub instance on 64-core workstation that was granted to me for use. I was then able to create an account for each student in the class, and sent them to the links and details on how to start a Jupyter notebook on their browser.

The next issue to resolve was the fact that most students do not have programming experience, and JupyterHub, though an excellent vehicle for learning Python, is very different from what they were used to in their concurrent Python course. I planned for a final computational project where I divided the students into 12 groups. I tried to distribute students with coding experience as evenly as possible. To bring the students up to speed, I employed a plan in which I awarded extra-credits to the students when they submitted their physics lab report in Jupyter Notebook. The first lab report I almost created the entire lab-report on Jupyter notebook myself, including the tables (using the Pandas Library). All they had to do was to plug in the numbers in the right part of the codes and then make the matplotlib plots. In the next lab-report I left more work for them in the notebook, and continued this trend where I kept reducing the amount of help I was providing to them. This turned out to be an excellent idea, because the students soon realized that this not only was good way of learning Jupyter, but also significantly reducing their work load, since much of the work in creating lab-reports follows the same templates, and that is handled much easily in Python.

Projects:

1. Projectile motion: Write a simple calculator tool for the motion of a particle launched at an angle w.r.t the horizon. The students used Euler's method to solve the kinematic equations of motion.

2. Collision problem: Inspired by the following video by [3Blue1Brown](), I asked the students to solve for the value of π by the method of elastic collisions. I was a little skeptical about the outcome, since this group did not have anyone who had a lot of prior coding experience. However, I was pleasantly surprised by the amount of hard work the students put, especially the team leader. After three meetings with her, she was able to write the entire code on her own to solve the problem.

3. Vector calculator: The goal was to have a tool that will compute the addition, subtraction, dot products, and cross product between vectors. The students wrote a very nice interactive code (not what I had in mind) that actually lets the users specify the number of vectors for which they will be conducting the operations. They were thorough in covering all the corner cases, such that the code does not fail.

4. Leaking mass problem: An Atwood machine had a mass on one end a leaky-bucket with water at the other end. The system was initially heavier than the counterweight. However, as water was leaking out, the mass was getting lower. The students had to first solve the equation of motion of this system analytically. The final part of the question was where the students will need to find out the time when the direction of motion of the bucket reverses. This requires solving a transcendental equation. I taught the students how to solve transcendental equations on a computer using the method of bisection. The project was incomplete in the end due to lack of time, but the student made good progress.g

5. Interceptor: This was a 2D kinematic problem where a target is launched with some velocity at some angle w.r.t the x-axis from the origin, and after a time an interceptor is launched from a distance on the x-axis with a velocity. The goal is to determine the minimum and the maximum time after the interceptor in launched for the interceptor to intercept the target. This range of time over which the interceptor can hit the target is given in the angle at which the interceptor is fired (with the extreme case of this angle being same as the angle of the target, when they will never meet). The students took some time to solve the analytical problem. They coded this up and created a nice calculator. However, due to lack of time the project was not finished, since the ultimate goal was to write a Monte Carlo simulation to test the analytic result.

6. Energy and momentum conservation problem: This was a one dimensional collision problem where a system of masses (each mass could be different) was given. The first mass is pushed with a known velocity and this will lead to a series of collisions, resulting in a collision-wave traveling from one end to the other. The students had to find the permutation of the masses that will result in the fastest propagation time of this collisional-wave. It turned out to be an easy problem in the end since in the first wave all the internal collisions were in the same direction.

7. Free-fall problem under: An object is attracted by another object due to gravity. The first object has a mass negligible compared to the second object. The goal is

to calculate the final kinetic energy of the collision. Students had to learn to write a time-evolution code using Euler's method to solve this and then compare using the analytical result.

8. Geo-stationary satellite: The goal was to calculate whether two points on a planet surface are visible by a satellite in a geostationary orbit and how long the signal takes to reach the satellite from both points. The result depends on the planet's mass, its radius and its rotation velocity. This problem is analytically solvable. The students had to solve the problem first using pen and paper, then had to write the codes necessary to convert this into a calculator.

9. Transit problem: The goal of this project was to model simplified orbits of planets and to calculate transit time for these planets. The actual problem is pretty difficult for this level. I simplified it for the student by stipulating that all the planets are located in the same plane. For the given number of planets the students had to find the time of the first alignment of all the planets since the last alignment. It proved to be an interesting numerical problem due to discreteness of time steps of the time-evolution code. The students needed some help from me to find out ways of capturing the actual points of alignments.

10. Moment of inertia: Given a mass distribution of objects the students have to find the moment of inertia of the system about an arbitrary (user-supplied axis).

11. Solving of radioactive decay equation using Euler's method: The original problem problem was to solve for the simple pendulum. However, this group had the least expertise in terms of computing knowledge. I had therefore made the problem easier, and hence changed it to a first order ODE.

12. Center of mass locator: Very similar to the moment of inertia calculator project. In this case the students had to write a code that will find the coordinates of the location of the center of mass of a non-trivial distribution of mass.

Conclusion:

It was a truly an enriching and challenging experience to teach this course, which was only possible due to the resources that were made available for me from XSEDE. I am planning to set up a website to share the notebooks from my students, at least in static form. Currently, I lack the resources maintain such a thing, hence I have put all of these on GitHub.