MATH198 Weekly Update

Professor George Francis
Brian Campbell-Deem

November 13$^{\text{th}}$ - December 11$^{\text{th}}$, 2015
Final Weeks ABCD

Abstract

Extreme forward movement to finish of project as time scope and circumstances allow. *hydrogen.py* has been built from scratch completely, simulating ensemble measurements of the ground state electron; *kinematics.py* has been reformatted to be user-interactive and repeatable. Bruce Sherwood's *gas.py* and *doublependulum.py* have been modified to fit the theme of the project as fits, allowing the user to enter different values and view how they affect the systems.

1 Preface

This update amalgamates several weekly updates into one. Due to unfortunate circumstances (e.g. laptop ceasing to function) both coding and update processes were staggered or halted altogether. The problems have been resolved and this update aims to succinctly describe the rather large changes to the project since then, which has been, for the most part, driven to the level of completion as described in the (rough proposal). Exact details of the project will be included in the finalized proposal, which includes the end-of-class narrative.

2 Code Progress

2.1 hydrogen.py

hydrogen.py has been built from the scratch as per the request of Professor Francis (proof of concept finished circa Nov. 20.) I used a brute-force computation of the integrated probability density to created weighted random distributions in the "measured" radius. After the radius is chosen, the angular components are chosen randomly with equal distribution (the probability distribution is angularly isometric). An array of points is initialized, and the animation loop constantly writes over their positions (with the aforementioned pseudo-random coordinates) with each new iteration. This creates an effect of a hazy cloud

which gives the general shape of the probability distribution. The animation runs inside of a box (the radii are limited to a certain, large and unlikely value), and the whole scene rotates via a method in the animation loop which constantly rotates the camera vector slowly.

2.2 kinematics.py

kinematics.py was mostly finished, however to follow my original plan more closely, I added the ability for the user to enter their own y velocity for the thrown object; they additionally may re-run the program with a new value through a loop method I included, allowing the different values to easily be compared visually. I also have a graph that updates in real-time with the animation displaying the kinematic properties of the thrown object, namely its acceleration, speed, and displacement from the origin.

2.3 gas.py (modified)

Due to time constraints, I chose to modify Bruce Sherwood's already excellent programs on the advice of Professor Francis. *gas.py* included pretty much everything I wanted, so I opted to clean it up a bit visually (graph titles and descriptions) as well as add the ability for the user to choose how many particles they want in the box. I added another graph, which also updates in real time, displaying the average collisions per second, which always decreases to some relaxing value after some large time; unfortunately I don't have time to include the theory on this value, but I chose to leave it in as another heuristic for the different equilibria the particles achieve.

2.4 doublependulum.py (modified)

doublependumlum.py is the other program of Sherwood's that I modified for my project. In this case, I essentially made the program run twice in two windows (with everything copied and renamed to avoid problems). I also added a sphere which is falsely attached to the end of the bottom pendulum to draw the path it follows, since an actual sphere on the bar is considered to not be moving by the program (the frame is what's moving). I added the ability for the user to toggle this path feature and pick different masses for the pendula; they also choose an angular offset to demonstrate the core purpose, which is how a small change can quickly lead to a completely different result in a chaotic system. The side-by-side views allow the user to quickly understand this visually.