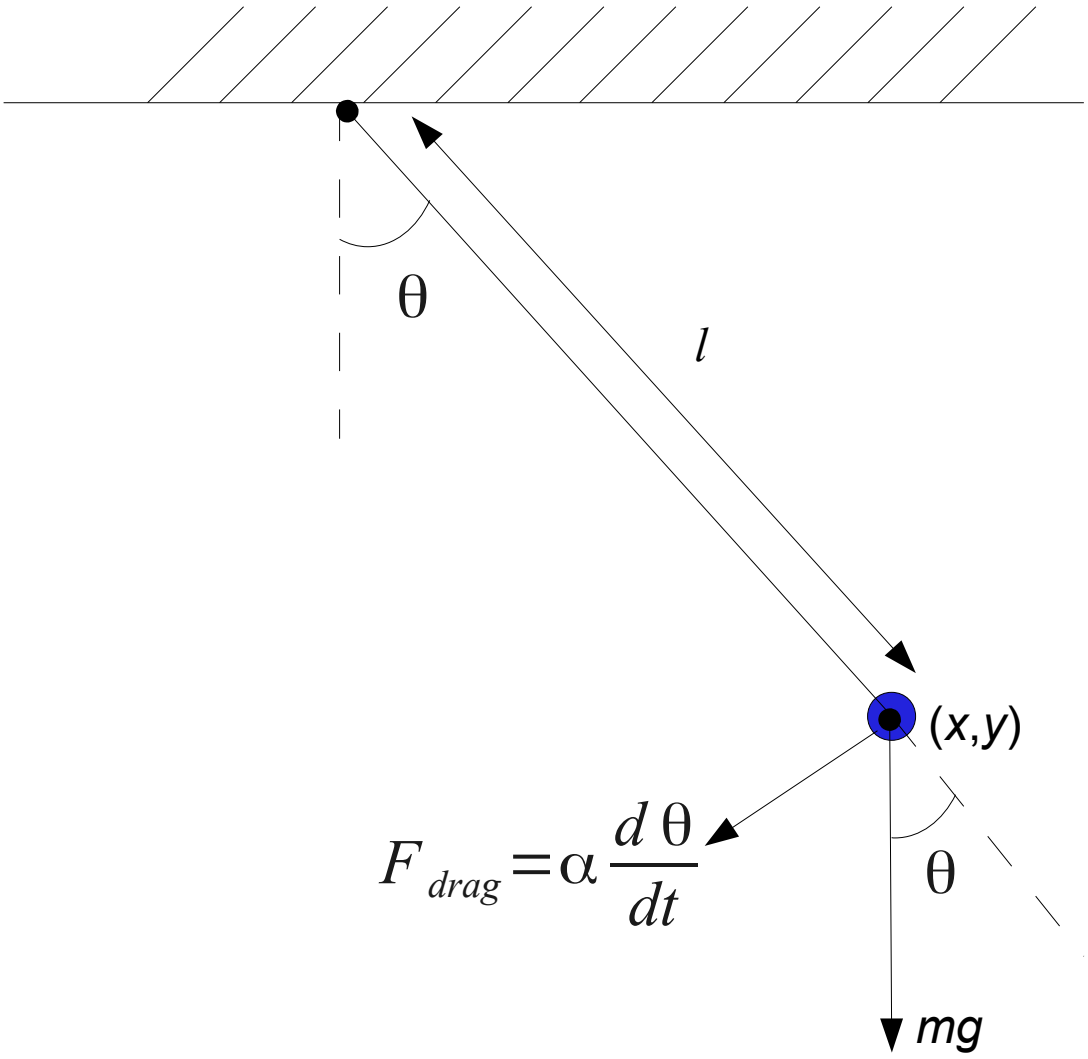
Physics 3340/8300-005 - "Computational Physics" - Fall 2013
Homework 11
Assigned: Monday, 25 November
Due: Friday, 6 December 11:59pm by Blackboard upload

Reading:

Cheney and Kincaid sections 7.1, 7.2, 7.4, 11.1 (6th ed.: 10.1, 10.2, 11.1, 11.2, 14.1)
http://www.physics.smu.edu/fattarus/DiffeqLab.html

Problems:
For the following problems submit your solutions in your upload, with any C source code you write plus enough text and gnuplot graphics to trace the process of your work.

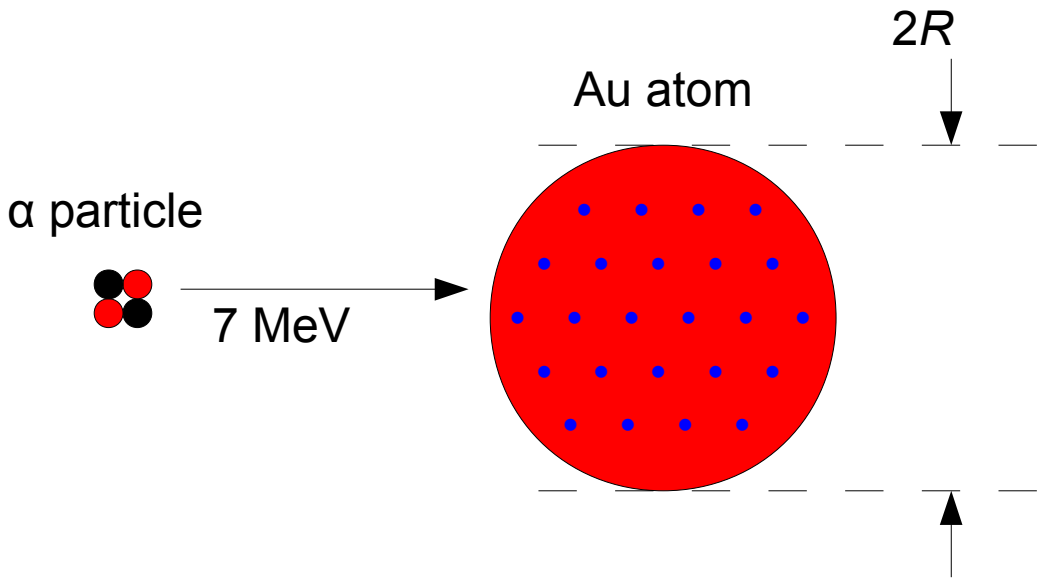
1) Simulate the forced damped pendulum as discussed in class with the Runge-Kutta algorithm. Write a C program that uses the `rk4()` routine in diffeq.c from the differential equations class lab. Your program may be patterned after the pendulum.c main program from that lab.



a) First simulate the case with no forcing (F=0) but with α=0.1 and some large initial value of the pendulum angle, such as 3 radians. Use l = 1m and run the simulation for 10 seconds. Create a gnuplot that shows the (initially) non-sinusoidal pendulum oscillation of the deflection angle with an amplitude decaying as a function of time and becoming more sinusoidal. Generate two plots, one with the angular displacement as a function of time, and the other a state-space plot with the angular velocity versus displacement.

b) Then turn both the initial states down to 0 and apply some sinusoidal forcing signal, and search for an operating condition that shows some chaotic behavior from the pendulum. Use α=0.05 and l = 1m, and run the simulation for 500 seconds. There are only some narrow regions in the multidimensional parameter space for a single arm pendulum that exhibits chaos, so some careful searching is required. One possible set of values to search for such behavior is setting the forcing amplitude F=4 and looking in the range of 2.5595 to 2.5597 radians per second for the forcing frequency, but you may also search for your own parameter set. Generate two gnuplots that show the state-space behavior of the system, that is, with the two state variables plotted against each other, completely changing behavior with very slight changes in a parameter value.

2) Simulate the Rutherford scattering experiment with the Runge-Kutta algorithm. Write a C program that uses the `rk4()` routine in diffeq.c from the differential equations class lab. Your program may be patterned after the ballistic.c main program from that lab.



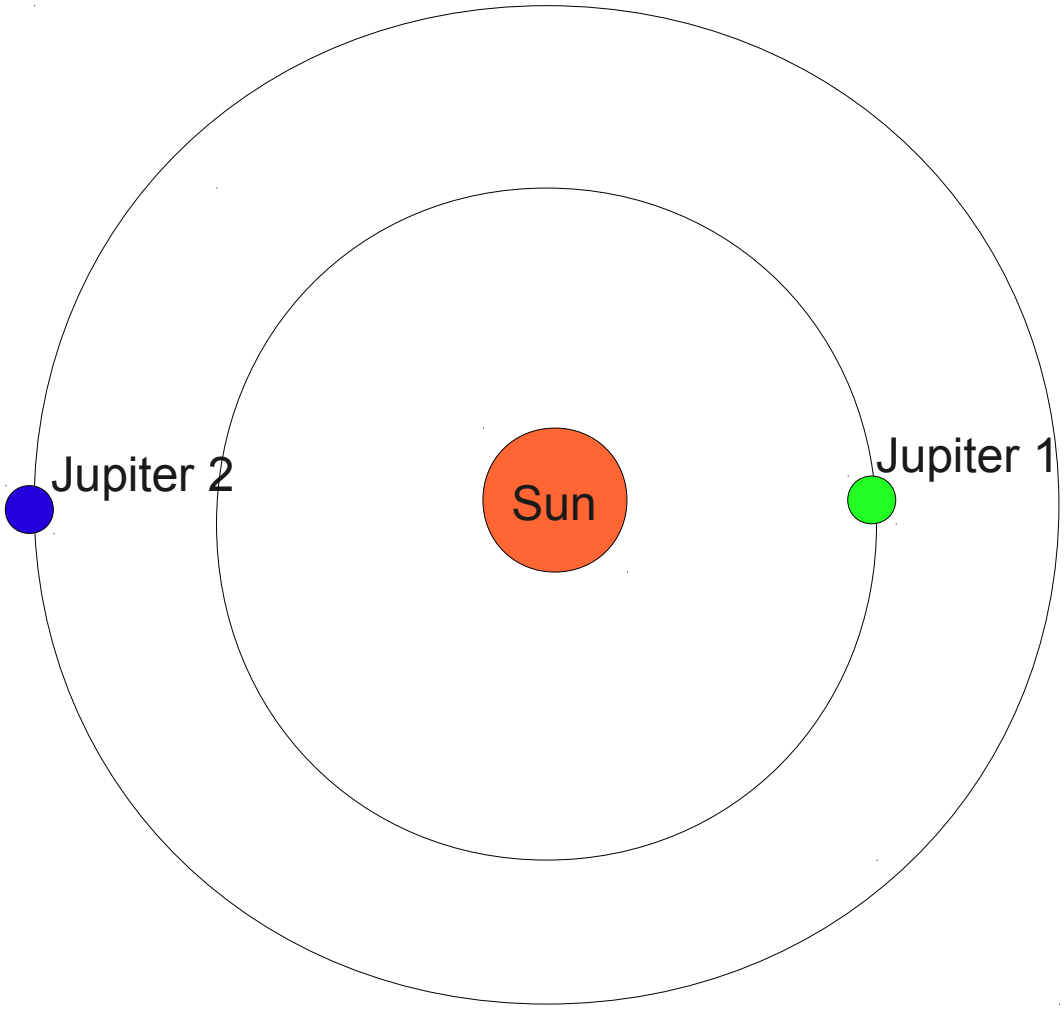
Start with the old assumption of an atomic model from J. J. Thompson, where the positive charge of the nucleus is uniformly distributed inside a sphere of radius R, where R was presumed to encompass the entire atomic size. Simulate the trajectory of an alpha particle incident from the left at an x velocity of 2×10^7 m/sec, which corresponds to about 7MeV kinetic energy. Place the center of the gold atom at x,y coordinates of (0,0) and start the simulation with the alpha particle at an initial x coordinate of -10R and

an initial *y* coordinate (the so-called impact parameter) of 0.5*R*. To simplify the simulation, assume the alpha particle is a point charge, and that the gold nucleus is so massive relative to the alpha particle that you may assume it is stationary. Also assume that the trajectory of the alpha particle stays within the *xy* plane, so only two dimensional equations of motion need to be considered. Run the simulation for a time duration sufficient to see the alpha particle be at an x coordinate of +10*R* if it were undeflected, that is, for a time required to travel a distance of 20*R* at a velocity of 2×10^7 m/sec. A time step value that lets the Runge-Kutta algorithm take 100 steps along the trajectory is sufficient.

Recall from electrostatic theory that if the charge distribution is assumed to be uniformly distributed inside a spherical volume, that the alpha particle will be subject to an electric field that may be calculated as if all the charge of the gold nucleus is located at a point at the center of the sphere. However, only the charge in the gold nucleus that is located at a radius less than the current radial location of the alpha particle will contribute to the electric field. Therefore the function you write to pass to the `rk4()` solving routine will have to calculate the electric field acting to deflect the alpha particle allowing for the two different cases of the alpha particle being outside or inside the sphere of charge.

Run your simulation code with three different assumed spherical radii, 10^{-12}m, 10^{-13}m and 10^{-14}m. Show that only the smallest nuclear size gives a sufficiently sharp deflection angle to account for the recoil events that Rutherford observed experimentally. Submit three gnuplots of your simulated trajectory for the three assumed nuclear radii.

3) Simulate with the Runge-Kutta algorithm the hypothetical solar system discussed in class with a sun identical to our sun being orbited by two Jupiter-like planets. Place the first planet in a circular orbit at the same radius as the Jupiter in our solar system, and the second planet in a circular orbit at 1.25 times that radius. Use 1.99×10^{30} for the mass of the sun, 1.90×10^{27} kg for the mass of each Jupiter, and an orbital radius of 7.78×10^{11} m for the inner planet. (Hint: This will require calculating two different velocities for the planets to initialize the simulation with, as the nominal orbital period for the two planets will of course be different.) Write a C program that uses the `rk4()` routine in diffeq.c from the differential equations class lab. Your program may be patterned after the ballistic.c main program from that lab. You may assume both orbits are in the same plane, so that the simulation may be limited to two dimensional motion. Hint: you will need four state variables for each gravitational body, for a total of twelve state variables in your system of differential equations.



a) Generate a gnuplot that shows the orbital trajectories of the two planets with a large gravitational interaction between the planets.

b) One of the current methods for locating exoplanets by astronomical observations relies on tracing the "wobble" a star experiences when orbited by massive planets. Generate a gnuplot that shows the "orbital" trajectory of the sun in this solar system.