PHYSICS 307 HOMEWORK 6

Due Tuesday, 30 October, at 5 PM

In this homework assignment, you will use your orbit program to simulate a highly elliptical orbits, and then some other things. If you don't finish by Tuesday, turn in what you have, and let Nick know when you'll have the finished product.

1. Consider a highly elliptical orbit such as a comet. At aphelion, Halley's comet is 35.1 AU from the Sun and has a velocity of 0.193 AU/year. Use your program to simulate the orbit of Halley's comet. How close does it come to the Sun at perihelion, and how fast is it traveling there?

   About how many timesteps are required to simulate this orbit reasonably accurately? Compare this to the value for a circular orbit that you determined

2. **Extra credit:** Can you think of any way to mitigate this and simulate the orbit with fewer timesteps? Implement it.

   last week. What makes cometary orbits so much harder to simulate accurately?

3. If you're not familiar with Kepler's laws of orbital motion, look them up. Does the behavior of your simulation reflect Kepler's second law (qualitatively; you do not need to measure areas!)

4. Now, modify your program to simulate a binary star system with two stars of unequal mass– that is, two objects responding to each other's gravity.

   Hints:

   - If you measure mass in solar masses, G still has a value of $4\pi^2$.
   - You will now need to use $\vec{F} = m\vec{a}$ and $\vec{F} = \frac{Gm_1m_2}{r_{12}^2}$ together to find the accelerations.
   - Your count of dynamical variables has now doubled to eight:
     - $x_1, y_1, v_{x_1}, v_{y_1}$ for the first star
     - $x_2, y_2, v_{x_2}, v_{y_2}$ for the second star
   - The leapfrog prescription is still the same:
     (a) Evolve the position variables (all four of them!) forward by $dt/2$
     (b) Evolve the velocity variables (all four of them!) forward by dt (this is the hard part)
     (c) Evolve the position variables forward by $dt/2$ again

     Your cut-and-paste skills will come in handy here, as you will be typing multiple copies of very similar code. But be careful of editing errors when doing this!

- The radius vector that appears in the differential equations is now the *separation vector* between the two stars. The origin no longer plays any special role in the dynamics.
- Since the x and y that appear explicitly in the differential equations come from the components of $\hat{r}_{12}$, they will become $x_1 - x_2$, etc.

5. Important: if you don't want your simulation to "drift" out of the viewport, then you will want to ensure that the total momentum is zero. See the end of this document for how to do this.

   Run your simulation, and monitor conservation of total energy to ensure that it is behaving properly. You now can't compute energy per unit mass, since you have two masses; the total energy will be

$$E = \frac{1}{2} m_1 v_1^2 + \frac{1}{2} m_2 v_2^2 - \frac{G m_1 m_2}{r_{12}} \tag{1}$$

6. Now, finally, modify your code to simulate the gravitational interactions of *three* bodies. This is no more complicated than two; you just have some copy-paste work to do, since each object now feels the force from *two* neighbors, rather than just one.

   Play around with what you can create – make things move in three dimensions, etc. Note that if two bodies get too close together, they will experience a very large force that is probably too big for your timestep to accurately simulate.

Then, I want you to do one (or both) of the following:

1 Option 1: Detection of exoplanets

Planets are small, close to stars, and don't give off their own light. They are very difficult to see directly with telescopes, although we can get some signatures if they go directly in front of their stars and block a bit of the light.

The easiest way to detect *exoplanets* – planets around stars other than our own – is to look for stars that are wobbling around a bit due to the influence of their *planets'* gravity. Even the gravity from the Earth causes the Sun to move. It turns out it is quite easy to detect changes in whether a star is moving toward or away from us using *Doppler spectroscopy*. Waves (light or sound) coming from an object moving toward us are shifted toward higher frequency (and thus bluer color); waves coming from an object moving away from us are shifted toward lower frequency (redder color). Gases in the atmospheres of stars emit and absorb very precise wavelengths of light based on the energy levels in the atoms; by very precise measurements of how these wavelengths shift around, we can use the Doppler effect to detect "wobbling stars" – ones that likely have planets. Modern spectrometers can detect radial motion of stars as little as one meter per second, with an extra factor of ten in sensitivity coming soon (0.1 m/s!)

Yep, that's right – we soon should be able to see stars hundreds of light years away moving at ten centimeters per second by looking very closely at their colors.

Simulate the following and look at the motion of the star. Plot one component of its velocity and determine the size of the fluctuations. Comment on (1) the possibility of detecting such a planet using this method, and (2) any interesting consequences you think such a detection would have for the possibility of extraterrestrial life.

- Earth orbiting the Sun. Earth has a mass of 3×10^{-6} solar masses.
- Jupiter orbiting the Sun. Jupiter has a mass of 0.001 solar mass (300 Earths) and orbits roughly 5 AU from the Sun. (You'll need to work out initial conditions for a nearly-circular orbit with radius 5 AU.)
- A "super-Earth" of mass 1.5×10^{-5} solar masses (5 times the mass of the Earth) orbiting Proxima Centauri, a small star with a mass of 0.12 solar mass, at a radius of 0.1 AU away

2 Option 2: Milankovitch cycles

The Earth's orbit is a little bit elliptical, but its eccentricity is not constant; instead, its eccentricity fluctuates over timescales of hundreds of thousands of years because of the gravitational influence of Jupiter (mostly) and Saturn. These fluctuations in eccentricity are responsible for the recent cycle of ice ages and interglacial periods, as I'll talk about in class.

Simulate the Earth, Sun, and Jupiter. (Earth has a mass of 3×10^{-6} solar masses; Jupiter has a mass of 0.001 solar mass and orbits about 5 AU away.)

Jupiter's orbit is a little bit elliptical and this is important for this simulation; to simulate this, set Jupiter's initial velocity to $2\pi/\sqrt{r_J} \times 0.95$. The first bit is the initial condition that gives a circular orbit; the factor of 0.95 controls the eccentricity.

Then, you'll need to modify your code to track the eccentricity of the Earth. The eccentricity depends on the aphelion and perihelion (furthest and closest distance to the Sun) – the wider the spread, the more eccentric. It took me a bit to figure out how to do this; I wound up using the following code that keeps track of the distance of the Sun on *three* successive timesteps to look for extrema, and prints out the eccentricity every hundred orbits. (You may copy this code, modifying it as you need.)

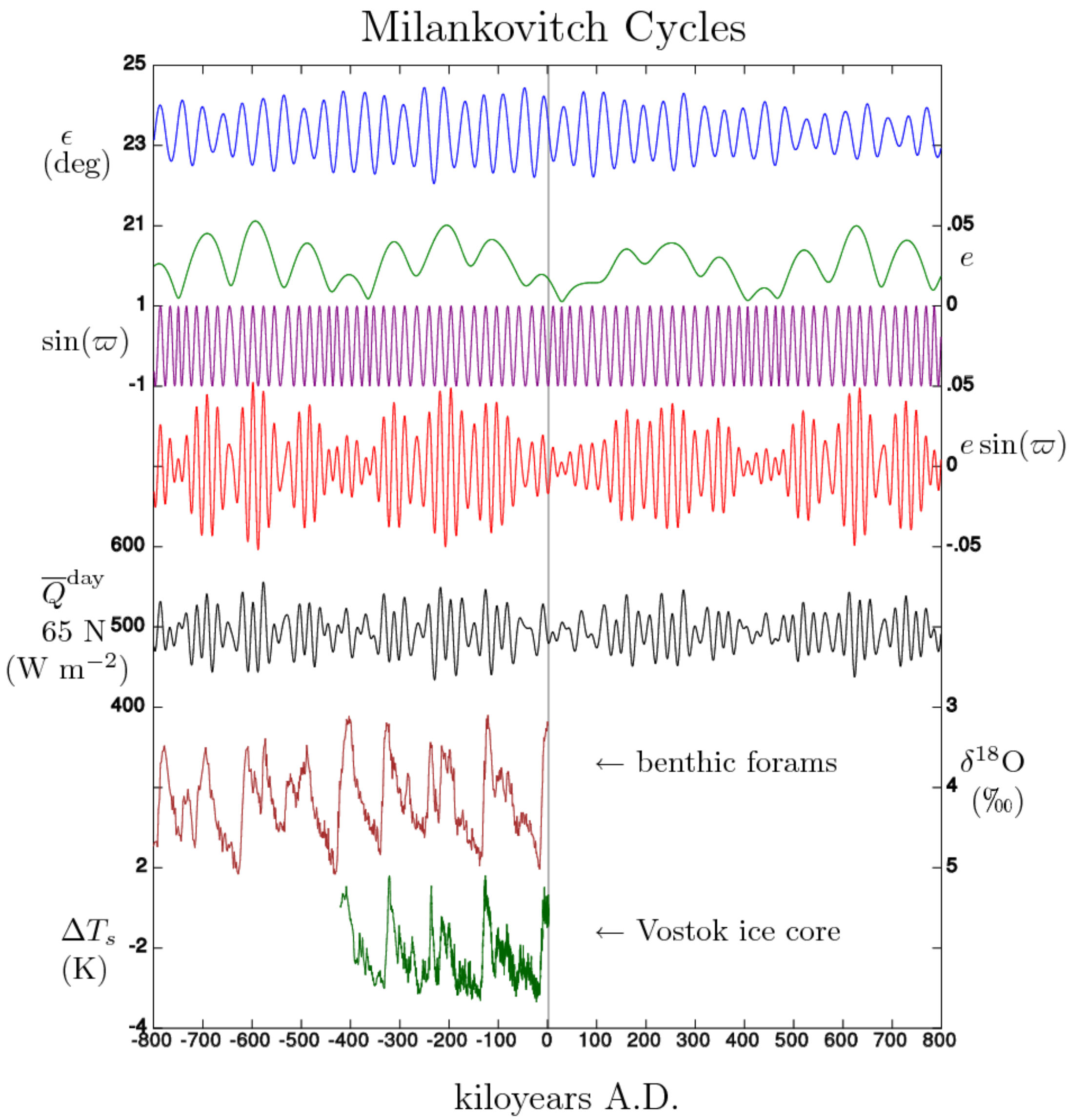
```
r_old_old=r_old;
r_old=r_now;
r_now=hypot(x1-x2, y1-y2);

if (r_old < r_old_old && r_old < r_now) // we are at perihelion
{
  r_peri = r_old;
}

if (r_old > r_old_old && r_old > r_now) // we are at aphelion
{
  aphs++;
  if (aphs % 100 == 0)
    eccentricity = (r_aph-r_peri)/(r_aph+r_peri);
    printf("!%e %e\n",t, eccentricity);
  r_aph = r_old;
}
```

One challenge for this project is that you will need to simulate many tens of thousands of years to see the eccentricity drift. You will need to use a small enoguh timestep to accurately simulate the small changes in eccentricity; I found that 10^{-4} years was sufficient. But you'll need to run it for a very long time; this means you'll need to run for quite a while to accumulate enough data. You'll need to either not animate your results or use a large frameskip to achieve this.

Show a plot of eccentricity vs. time like the one I showed in class. Then compare this to the actual fluctuations of the Earth's eccentricity, the first green trace on the plot on the next page. Are they similar in character? How do they differ? Look at the Wikipedia page on "Milankovitch cycles" for background if you want.



Milankovitch Cycles
ϵ (deg)
e
$\sin(\varpi)$
$e\sin(\varpi)$
$\overline{Q}^{\text{day}}$ 65 N (W m^{-2})
← benthic forams
δ^{18}O (‰)
ΔT_s (K)
← Vostok ice core
25
23
21
1
-1
600
500
400
2
-2
-4
.05
0
.05
0
-.05
3
4
5
-800 -700 -600 -500 -400 -300 -200 -100 0 100 200 300 400 500 600 700 800
kiloyears A.D.

Hints: Avoiding solar system drift

The total momentum of a system of two stars is

$$\vec{p} = m_1\vec{v}_1 + m_2\vec{v}_2$$

This gives a center-of-mass velocity of

$$\vec{v}_{\mathrm{com}} = \frac{m_1\vec{v}_1 + m_2\vec{v}_2}{m_1 + m_2}$$

By calculating this value and subtracting it from each of your objects' initial velocities, you ensure that the total center-of-mass velocity (and thus the total momentum) is zero, and your simulation won't drift. This can be done with code like the following:

```
double vxc,vyc; //center-of-mass velocities
vxc = (m1*vx1 + m2*vx2) / (m1+m2);
vyc = (m1*vy1 + m2*vy2) / (m1+m2);
vx1 = vx1 - vxc;
vy1 = vy1 - vyc;
vx2 = vx2 - vxc;
vy2 = vy2 - vyc;
```