Physics 350 – Undergraduate Classical Mechanics
Numerical Homework II,
due Friday, September 23, 2011 at 11:00 a.m.

Note: This homework has three pages.

All solutions must be completely typed – no hand-written material will be accepted.
Also, hard copies only – no electronic submissions this time.

In our first attempt to drop a Mars probe onto Mars (Numerical HW I), we didn't take into account a few things. We didn't say how we were going to "drop" the probe. It will be dropped from a satellite that will be orbiting Mars in a polar orbit (http://en.wikipedia.org/wiki/Polar_orbit). The satellite will be at some height above the surface in a circular orbit. If we launch the probe backwards with the same speed as the satellite, it will fall straight down. However, the launcher on the satellite can only launch the probe at a maximum of 400 m/s relative to the satellite. So the question is this: if you release your probe when the satellite is at the equator in it's polar orbit, with what speed should you launch your probe so that it hits directly at the north pole? To make it simple, let's say that the launcher launches the probe tangential to the orbit (no initial radial component of the velocity).

What you know is that the satellite will be orbiting in a circular orbit. Again, you can assume the 10 kg mass in your Mars probe is evenly distributed, and it has a radius of five meters. As before, you should include in your estimate the drag due to Mars' atmosphere and the changing value of the local gravitational acceleration as the probe approaches the surface.

Recall that the gravitational force is

$$F_{\text{grav}} = -\frac{GmM}{r^2}\hat{e}_r \, . \tag{1}$$

You can take the force the Martian atmosphere exerts on the probe to be

$$F_{\text{atmo}} = -\frac{1}{2}C_d\,\rho_{\text{atmo}}(r)Av^2\hat{e}_v \tag{2}$$

with r the height above the surface, $C_d = 0.2$ the dimensionless drag coefficient, and A and v the cross-sectional area and velocity of the probe. Note that $\hat{e}_v$ is the unit velocity vector. The local atmospheric density is

$$\rho_{\text{atmo}}(r) = \rho_0 \exp(-r/r_0)\, . \tag{3}$$

Some thought/analytical work you need to do before you start: You could do this problem in cartesian coordinates, but it lends itself to use polar coordinates. Set the problem up in polar coordinates. I showed you in Lecture 4 what the radial and angular acceleration looked like in terms of r and θ. What you want to use is the sum of the forces in the radial and angular direction from that lecture.

Use those equations to solve for $\ddot{\theta}$ and $\ddot{r}$. Then you numerically integrate (using the shooting method again) to get $\dot{\theta}$ and $\dot{r}$ and finally θ and r. Note that gravity will only point radially, but the air drag will have a component in each direction.

You can also solve for things in your loop like v and a at each time. One thing that you will have to do is to plot the actual orbit of the probe in some different situations. To do this you will have to make a table that has x,y pairs in it, so when you list plot, it will show you all the locations where the probe has been (i.e. the trajectory). Remember how to relate r and θ to x and y?

To solve this problem you will need to look up the mass and mean radius of Mars, as well as determine the mean surface level atmospheric density ρ_0 and and atmospheric scale height r_0. Take a look at NASA's website, http://nssdc.gsfc.nasa.gov/planetary/factsheet/marsfact.html.

1. Here's how I would proceed with the programming: Make a Module in Mathematica that takes the following inputs (it's important that it can take all of these inputs!):
   (a) The maximum time that you will allow the simulation to run
   (b) The time step that you will use in your calculation
   (c) The height of the circular orbit that you will be dropping from
   (d) The tangential velocity that you will be launching the probe from the satellite.
2. The module should do the following things. All of this should be inside your module, so that you can just run the module with different inputs and it will give you the answer.
   (a) It should find the speed of the satellite (this is easy because you know the force due to gravity and you know that that must equal the centripetal acceleration of the satellite times its mass; note you don't need to know the mass of the satellite).
   (b) Initialize some tables that have enough elements to make it to your maximum time. Some people rather than doing this step, were using AppendTo. This is ok, but AppendTo runs so much slower that you might regret it in this assignment.
   (c) Set your initial conditions.
   (d) Go through a loop that will find the position, velocity and acceleration at each time step and tabulate them in your tables. Note that you have two directions now that you have to keep track of. The loop should exit when the probe hits the ground and tell you what angle has been traversed. If you launch at the equator, and you want to hit the north pole, the final angle should by $\pi/2$.
   (e) The module should return the angular displacement where it hits Mars! This is important for the last part of this assignment.
   (f) When you think you're finished, as a first check, I would redo the problem from last time (I won't grade for this though, I just think it's a good idea). The way you do this is launch the probe directly backward with the velocity of the satellite. This will cause it to fall straight down. If you get results that are pretty much the same (taking into account that there might be some numerical difference), that's a good sign that things are working.
3. Once you get it to that point, here is the stuff we will grade (outside of the base code that you need to do this):
   (a) As a new check, see what happens if you just let the probe go from a satellite that is in a circular orbit 1000km above the surface. What you should see is that it doesn't fall. It goes around in a circular orbit with your satellite. Plot this orbit as proof that you did it.
   (b) Now check to see what happens if you launch your probe in the wrong direction (from the same satellite). Launch it with 400 m/s in the same direction the satellite is traveling. It should go in a slightly elliptical orbit around the planet. Note, that both this orbit (and the last circular one) should be closed! That means that if it doesn't come pretty

much right back on itself in both cases, something is wrong in your code. This is also a good check to make sure you have it set up right. Make sure that your maximum time is set up right so that it won't quit before it closes the orbit. Plot this orbit as proof you did it.

(c) Now actually do the problem. Find the tangential speed you want to launch your probe (from the same satellite) when the satellite is right above the equator so that the probe will crash right at the north pole. Give the speed to the nearest m/s, and plot the orbit of the probe to prove that you did it. There are elegant ways of finding solutions like this, but for now, just run your module and find it by trial and error. (For a ball-park, I get something around 350 m/s; this is your last check)

(d) Say that the satellite is now at 2000km high in a circular orbit. At what speed would you have to launch the probe backwards in order to hit the north pole? Also plot the orbit.

(e) Say that the satellite is now at 4000km high in a circular orbit. At what speed would you have to launch the probe backwards in order to hit the north pole? Also plot the orbit

(f) Say that for the satellite at 1000km height, your launcher jams so that you can't control the speed that it's going to launch, and it will launch at 400 m/s (opposite the direction the satellite is moving again). At what latitude should you launch in order to hit the north pole?

(g) I used a time step of 1 second for most of my calculations. That seemed sufficient, but how can we tell? Part of good numerical practice is understanding how much your discretization has affected your answer. Using the solution to part (c), plot the angular displacement ($\Delta\theta$) as a function of time step. Note that the answer should approach $\pi/2$ as the time step gets small. To do this, make a table that contains some dt's ranging from 1 to 20 in steps of 1 that you will plug into your module. Run the optimization module (using another Do or For loop), and plot the angular displacement as a function of dt. Does it looks like the curve limits to a number as dt$\rightarrow$0? What is the percent error between your result using dt=1, and the limit as dt$\rightarrow$0?