2019

09 Two Capstone Projects

W.A. Morgan
Dickinson College

L.Q. English
Dickinson College

Two Capstone Projects

9.1 The Gravitational "Slingshot"

In the previous chapters of this book you have gone from investigating the simple motions of an object with constant velocity to exploring the physics of a comet about the Sun. While these phenomena are "real life", we want to end with a consideration and discussion of a truly "Space Age" phenomenon that is used regularly in sending space probes to the outer solar system – **gravitational assist**, or the "slingshot" maneuver.

The outer planets – Jupiter, Saturn, Uranus, and Neptune – as well as the objects in the Kuiper Belt – including Pluto, Makemake, and $2014\ \mathrm{MU}_{69}$ – are extremely far away. Jupiter is just over 5 astronomical units[1] away from the Sun, Neptune is 30 a.u. away from the Sun, and Pluto is 40 a.u. away from the Sun. Traveling to any of these objects will take several years to get there.

But *how* to get there? Traveling directly away from the Sun to arrive at a distant planet would take an extraordinary amount of fuel to travel – and that extra fuel means more mass, and the more mass, the more fuel needed. It would be extremely costly, and at this writing we do not have the technology to develop rockets to transport a space probe those distances. The Hohmann orbit (least energy method) saves on fuel, but takes quite a while to complete the journey.

Gravitational assist allows a space probe to take some of the energy from a nearby planet and use it for itself. The energy is "free"!

[1] An astronomical unit is the average distance of the Earth to the Sun. It is equal to 1.50×10^{8} kilometers.

9.1.1 The Physics

The slingshot maneuver involves a planet and a space probe. Both objects have a velocity (therefore, both a speed and a direction). When the probe nears the planet and is appreciably affected by the planet's gravity, it accelerates (both in speed and direction). The interaction is a "collision", but generally nothing collides. Furthermore, the collision is elastic, so therefore there is no loss of energy (and, of course, no change of total momentum) in the planet-probe system.

So, if the kinetic energy of the probe increases, then the kinetic energy of the planet *d*ecreases. The planet slows down! However, the amount of decrease of kinetic energy is negligible to the planet, because the mass of a planet is **huge** compared to the mass of a space probe. For example, the mass of Uranus or Neptune is approximately 10^{26} kg, compared to the mass of a space probe, which is typically on the order of 10^3 kg.

In the rest frame of the Sun, the Sun "sees" the planet moving with initial velocity $\vec{U}_i$. The space probe is moving with initial velocity $\vec{v_i}$. The two objects are not necessarily going to meet head-on; that would certainly result in the loss of the spacecraft. Instead, their velocity vectors are separated by a distance b, called the **impact parameter**. If $\mathrm{b} \leq \text{the radius of the planet } \mathrm{R_p}$, then there *would* be a head-on collision.

Consider a planet and a space probe. In the Sun's frame of reference, the probe has velocity $\vec{v}$ and the planet has velocity $\vec{U}$ (see Figure 9.1). Therefore the probe's kinetic energy is $\frac{1}{2}mv^2$. In the planet's frame of reference, the planet sees the probe approach with speed $v + U$. As it gets closer, the speed of the probe increases, and after the encounter, the probe's speed decreases until it appears to the planet to have a speed $v + U$ again, as shown in Figure 9.2. (We are

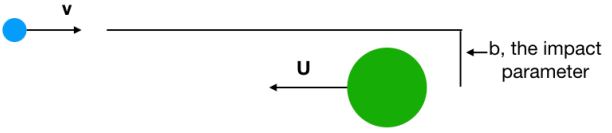


Figure 9.1: Space Probe Approaching Planet

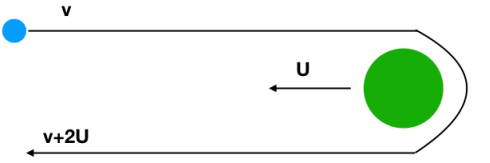


Figure 9.2: Space Probe After Experiencing Gravitational Assist From Planet

assuming an ideal encounter that causes the probe to make a 180° change of direction). Therefore, in the Sun's frame of reference, it appears as if the probe has speed $v + 2U$.

In the Sun's frame of reference, then, the kinetic energy of the probe is $\frac{1}{2}m(v + 2U)^2$. The increase of energy to the probe is $2mU(v+U)$. (By the same token, because the energy is conserved in this elastic collision, the planet's kinetic energy goes down by the same amount, but the change is negligible for the planet).

Several probes to the outer Solar System have used gravitational assist. They include the *Cassini* mission to Saturn (see Figure 9.3), the *New Horizons* mission to Pluto and 2014 MU_{69}, and the famous

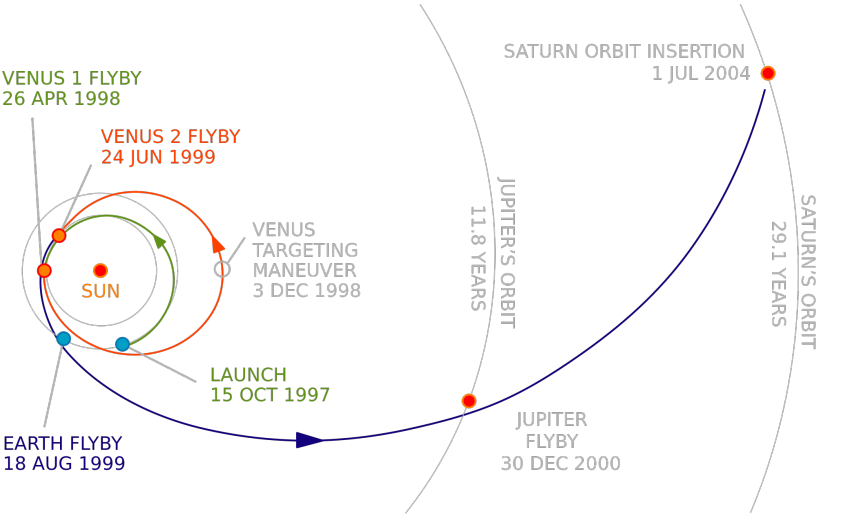


Figure 9.3: Trajectory of the *Cassini* Mission to Saturn (source: NASA)

probes *Voyager* 1 and 2, which are now in interstellar space.[2]

9.1.2 The Code

In Figure 9.4 the kinetic energies of the probe and the planet are calculated (e.g., in line 11) as $\frac{p^2}{2m}$, which is the same as $\frac{1}{2}mv^2$. In line 27, the gravitational force is calculated, and lines 34 and 35 update the planet's and the probe's positions at each instant.

[2] *Voyager* 1, launched in 1977, is now almost 150 a.u. from the Sun and is the farthest human-made object in existence.

```
GlowScript 2.8 VPython

scene.range=2e9

G = 6.7e-11

planet = sphere(pos=vector(2e8,4e8,0), radius=5e7, color=color.red,
                make_trail=True, interval=10)
planet.mass = 1e26
planet.p = vector(-1.5e3, 0, 0) * planet.mass
planet.K = mag2(planet.p)/(2*planet.mass)

probe = sphere(pos=vector(-2.0e9,5.2e8,0), radius=1e7, color=color.yellow,
                make_trail=True, interval=10)
probe.mass = 1e4
probe.p = vector(5e3, 0, 0) * probe.mass
probe.K = mag2(probe.p)/(2*probe.mass)
print("Initial KE of Probe:", probe.K, ".  Initial KE of Planet", planet.K)

dt = 2e2
r = planet.pos - probe.pos
r0=r

while mag(r)<=mag(r0):
    rate(300)
    r = planet.pos - probe.pos
    F = G * planet.mass * probe.mass * r.hat / mag2(r)

    planet.p = planet.p - F*dt
    probe.p = probe.p + F*dt
    probe.K = mag2(probe.p)/(2*probe.mass)
    #print(satellite.K)

    planet.pos = planet.pos + (planet.p/planet.mass) * dt
    probe.pos = probe.pos + (probe.p/probe.mass) * dt
print("Final KE of Probe:", probe.K, ".  Final KE of Planet", planet.K)
```

Figure 9.4: Sample Code for Gravitational Assist

9.1.3 Exercises

1.
   - Run this program to make sure it works. Describe what you see.

   - Modify the code to incorporate an impact parameter, b.
   - What happens when you change the impact parameter, to make it larger or smaller?

   - What happens when you change the velocities of the probe or the planet? In particular, what happens to the speed of the probe when the planet's velocity is 0?

   - Similarly, what happens when the planet's velocity changes sign? In that scenario, the probe and the planet are initially moving in the same general direction, with the probe catching up to the planet.

2.
   - Now that you have explored the b-dependence a bit, let's make this a bit more precise. Choose five different impact parameters and keep track of the angle of deflection of the probe for each. Enter the values for b and deflection angle into a spreadsheet and plot the relationship between those variables. If the graph is not clear, go back and add a few more b-values into your table.

   - For the same b-values that appear in your list, write down the the change in kinetic energy of the probe, expressed as a fraction of the initial kinetic energy, based on the code output. Then enter these values for fractional change in kinetic energy into your spreadsheet and plot its dependence on impact parameter.

Based on this plot, if we want a large increase in kinetic energy for the probe, what do we have to do?

9.2 Asteroid near a Binary Star System - Chaotic orbits, or the "Three-Body Problem"

In the previous chapters you have gone from investigating the simple motions of an object with constant velocity to exploring the physics of a comet about the Sun. In all of these examples, the motion did not exhibit what physicists call *deterministic chaos*. In the gravitational problem of two bodies attracting one another, solutions can, in fact, be found by pencil and paper. Interestingly, when we add a third body all bets are off. The orbits can no longer be predicted with a mathematical formula, but worse than that, the orbits exhibit what is known as *extreme sensitivity to initial conditions* - a hallmark of chaos.

9.2.1 Modifying a previous code

To keep things as simple as possible, let us revisit an earlier problem - the binary-star system in Section 7.1. The question we want to ask now is: if we introduced a small asteroid into this system how would it move under the influence of the gravitational attraction to each star? To make the problem even simpler, we assume that the asteroid is so small that it does not affect the motion of the two stars. Since it is 8 orders of magnitude (or 100 million times) less massive than the stars, this is a very good approximation. In that case, the mass of the asteroid really doesn't enter into the picture. All we need to do is to calculate the acceleration experienced by the asteroid at

every time step, and this is given by the universal law of gravitation as,

$$\vec{a} = -G\frac{m_1}{r_1^2}\hat{r}_1 - G\frac{m_2}{r_2^2}\hat{r}_2, \tag{9.1}$$

where $\vec{r}_1$ is the vector from first star to the asteroid, and $\vec{r}_2$ is the vector from the second star to the asteroid, and the m's in the numerator are the star masses.

Once we have computed the accelerations, we can update the asteroid velocity, and then we use the updated velocity to find the next position of the asteroid, in a recursive procedure outlined in Chapter 6.

So here is the idea. Let us start with the code in Figure 7.1 which will take care of the motion of the two stars. To give us a little more "room", let's start these two stars out along the x-axis at -5e11 and +5e11 (instead of $\pm$2e11. Also, to keep the center of mass of the system fixed, let's adjust the y-component of the small start from -1e4 to -9e3. Finally, turn off the trails on these two stars, and delete all calculations of the center of mass (we won't need that anymore). This takes care of the binary-star system.

Now let's introduce the asteroid. Add a line defining this object near the top that defines the asteroid as a sphere. Give it an initial position of `vector(1.50e12,0,0)`. Then define a asteroid velocity as one of its attribute as
`asteroid.v = vector(0,-7.5e3,0)`.

Now that we have defined the asteroid and its initial position and velocity, we need to turn our attention to its motion. At the end of the while-loop, add two lines that compute the vectors $\vec{r}_1$ and $\vec{r}_2$, appearing in Equation 9.1. Each of them are just the differences in the position vectors of the asteroid and the respective star. With these two vectors in hand, we can compute the asteroid's acceleration, `asteroid.a`, using Equation 9.1 directly.

```
if (mag(r1)<2e9 or mag(r2)<2e9):
    print("collision")
    break
elif (mag(r1)<8e9 or mag(r2)<8e9):
    dt = 4e3
    print("close call")
elif (mag(r1)<8e10 or mag(r2)<8e10):
    dt = 2e4
else: dt = 1e5
```

Figure 9.5: A rudimentary adaptive time step

This program would now run, but it is a good idea to add two more things. The first one is straightforward. If the asteroid comes too close it burns up in the stellar atmosphere. So we should include an if-statement to terminate the program if the asteroid comes within, say, `2e9` (or 2 million kilometers) of either star.

The second refinement has to do with the computational integration technique itself. Right now we are using a simple Euler-Cromer algorithm, as explained before. But you may have noticed that when the asteroid gets really close to either star, the dynamics is no longer smooth. The asteroid picks up so much speed that it covers a significant amount of space between successive time steps. In other words, the computational time interval, dt, seems to be too large, and the result is no longer accurate.

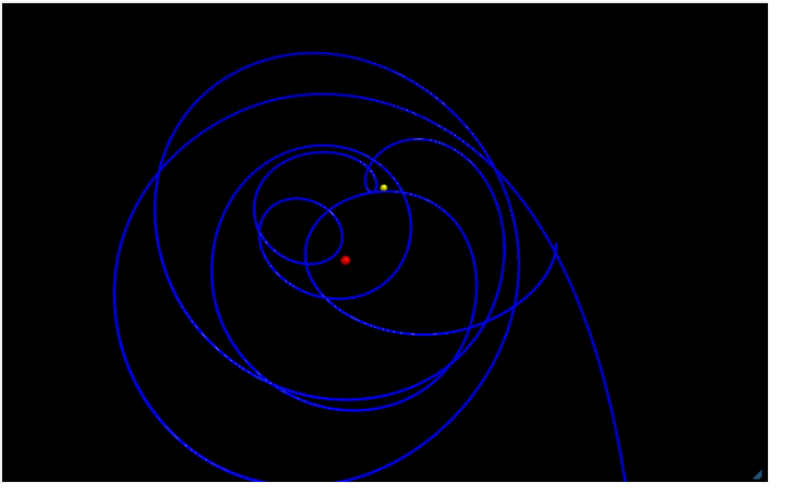
In a more advanced course, you would now probably be learning about the Runge-Kutta algorithm with an *adaptive time-step*. But that would go beyond the scope of this course. So, instead, let's implement a most rudimentary version of an adaptive time-step. The basic idea is that when the asteroid comes within a certain radius of either star, then we need to compute more finely by reducing the dt.

Figure 9.5 shows one simple way we can accomplish this. Keep in

mind, however, that this is not a highly accurate algorithm either, and that there are much more sophisticated ways to do this out there. Nonetheless, it improves on our previous method.

9.2.2 Exercises

Now that you have completed the coding, let us have fun and run the program with a few initial conditions. You should get output like the following:



One thing that is immediately clear is that the orbits are highly irregular and non-periodic. Furthermore, what you can do is to vary the initial position of the asteroid slightly. For instance, you can check the orbits that result from a initial x-coordinate of 1.49, 1.50, and 1.51×10^{12} meters.

Find initial conditions that lead to the three possible long-term outcomes for the asteroid: (a) it crashes into one of the stars, (b) it

escapes the binary-star system and drifts further and further into space, and (c) it settles into a semi-stable orbit around one of the stars.