Computer exercise: Bridges and tails

In this computational exercise we study how galaxy interactions are responsible for creating bridges and tails of observed galaxies (see Figure 1). First, it is explained how the 2-body problem is solved numerically, and second a simple model shows how a galaxy disk responds to a merger.

Figure 1: Upper figure: NGC4676: An observed galaxy merger where a tail and a bridge has been created by tidal interactions. Lower figure: A bridge is present in the merger, Arp 220. Explanation from APOD 28.11.2016: Why is there a bridge between these two spiral galaxies? Made of gas and stars, the bridge provides strong evidence that these two immense star systems have passed close to each other and experienced violent tides induced by mutual gravity. Known together as Arp 240 but individually as NGC 5257 and NGC 5258, computer modelling and the ages of star clusters indicate that the two galaxies completed a first passage near each other only about 250 million years ago. Gravitational tides not only pulled away matter, they compress gas and so caused star formation in both galaxies and the unusual bridge. Galactic mergers are thought to be common, with Arp 240 representing a snapshot of a brief stage in this inevitable process. The Arp 240 pair are about 300 million light-years distant and can be seen with a small telescope toward the constellation of Virgo. Repeated close passages should ultimately result in a merger and with the emergence of a single combined galaxy.

Part I: Solving the two-body problem

Initial conditions

We will solve the 2-body problem numerically for a planet (B) orbiting a star (A). We work in units with $G = 1$ and let the masses be $M_A = 1$ and $M_B = 10^{-6}$. As initial

coordinates we take,

$$\begin{aligned}\mathbf{x}_A &= (0,0,0),\\ \mathbf{x}_B &= (1,0,0),\\ \mathbf{v}_A &= (0,0,0),\\ \mathbf{v}_B &= (0,\sqrt{GM_A/x_B},0).\end{aligned}$$

This corresponds to the planet being in a circular orbit with a radius of 1.

The Euler and Leapfrog integrators

To integrate the system we use two different integrators; the Euler and the Leapfrog integrator. For a simple explanation of how to integrate a N-body system in time see section 9 (especially section 9.6) in Volume I of the Feynman lectures: `http://www.feynmanlectures.caltech.edu/I_09.html`.

At the initial time, $t = 0$, we know the position, $\mathbf{x}$, and velocity, $\mathbf{v}$, of the particles in the simulation. From this the acceleration, $\mathbf{a}$, can be calculated using Newton's laws.

The simplest (but not most accurate!) method to integrate a N-body system is the Euler method, where the first timestep can be written as

$$\begin{aligned}&\text{Evaluate } \mathbf{a}_0 \text{ based on } \mathbf{x}_0,\\ \mathbf{x}_1 &= \mathbf{x}_0 + \mathbf{v}_0\Delta t,\\ \mathbf{v}_1 &= \mathbf{v}_0 + \mathbf{a}_0\Delta t.\end{aligned}$$

Here a subscript indicates the number of a timestep, i.e. 0 corresonds to $t = 0$ and n corresponds to $t = n \cdot \Delta t$. With these three operations we have hence evolved the system from $t = 0$ to $t = \Delta t$. By repeating this procedure multiple times, the system can be evolved for an arbitrary amount of time. A general way of performing a timestep is

$$\begin{aligned}&\text{Evaluate } \mathbf{a}_n \text{ based on } \mathbf{x}_n,\\ \mathbf{x}_{n+1} &= \mathbf{x}_n + \mathbf{v}_n\Delta t,\\ \mathbf{v}_{n+1} &= \mathbf{v}_n + \mathbf{a}_n\Delta t.\end{aligned}$$

It can be proven that this method is accurate to first order.

In the Leapfrog algorithm a higher accuracy (2nd order) is obtained by making a velocity offset at the initial time:

$$\begin{aligned}&\text{Evaluate } \mathbf{a}_0 \text{ based on } \mathbf{x}_0,\\ \mathbf{v}_{-1/2} &= \mathbf{v}_0 - \mathbf{a}_0\Delta t/2.\end{aligned}$$

After the initialisation the first timestep can be performed:

$$\begin{aligned}&\text{Evaluate } \mathbf{a}_0 \text{ based on } \mathbf{x}_0,\\ \mathbf{v}_{1/2} &= \mathbf{v}_{-1/2} + \mathbf{a}_0\Delta t,\\ \mathbf{x}_1 &= \mathbf{x}_0 + \mathbf{v}_{1/2}\Delta t.\end{aligned}$$

This can be repeated and the system can be integrated for an arbitrary long time. The n'th timestep takes the form:

$$\begin{aligned}&\text{Evaluate } \mathbf{a}_n \text{ based on } \mathbf{x}_n,\\ \mathbf{v}_{n+1/2} &= \mathbf{v}_{n-1/2} + \mathbf{a}_n\Delta t,\\ \mathbf{x}_{n+1} &= \mathbf{x}_n + \mathbf{v}_{n+1/2}\Delta t.\end{aligned}$$

Figure 2 illustrates how the position and velocity for a particle are evolved with the leapfrog integrator.

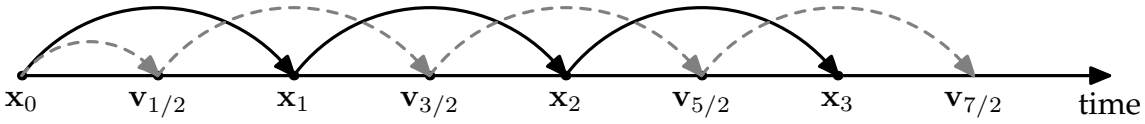

Figure 2

- Run the program, **BridgesAndTails_PartI.py**, which integrates the above two-body problem with the Euler and the Leapfrog method. Initially you can use a Δt corresponding to 5% of an orbital time, $2\pi x_B/\sqrt{GM_A/x_B}$, and run the simulation for 10 orbital times. Show that the Leapfrog method gives a more accurate integration of the circular orbit than the Euler method.
- Show that both methods converge to a circular orbit when Δt is sufficiently small.

Part II: Inclusion of a stellar disk

Initialisation of galaxy orbits

Instead of being a star and a planet, A and B are now two galaxies. They are initialised as follows:

$$
\begin{aligned}
M_A &= 1 \\
M_B &= 0.3333 \\
\mathbf{x}_B &= (15, 40, 0), \\
\mathbf{x}_A &= (-M_B x_B/M_A, -M_B y_B/M_A, 0.0), \\
v_{\text{esc}} &= \sqrt{\frac{2GM_A}{|\mathbf{x}_A - \mathbf{x}_B|}} \\
\mathbf{v}_B &= (0, -v_{\text{esc}}, 0), \\
\mathbf{v}_A &= (0, \frac{M_B}{M_A} v_{\text{esc}}, 0).
\end{aligned}
$$

The initial speed is selected to be the escape speed at the initial distance. An offset has been added in the x-direction, so the galaxies collide with an impact parameter. The velocities are chosen, so we work in the center-of-mass frame. The python code for the initialisation looks like this:

```
M0 = 1.0
M1 = 0.333
X1,Y1,Z1 = 15.0,40.0,0.0
X0,Y0,Z0 = -M1/M0*X1,-M1/M0*Y1,0.0
Vesc = numpy.sqrt(2*G*M0/numpy.sqrt( (X0-X1)**2 + (Y0-Y1)**2 + (Z0-Z1)**2 ))
VX1,VY1,VZ1 = 0.0,-Vesc,0.0
VX0,VY0,VZ0 = 0.0,-M1/M0*VY1,0.0
```

The program will produce several output images between 30% and 90% of the infall timescale,

$$t_{\text{infall}} = \frac{\pi}{2}\frac{|\mathbf{x}_A - \mathbf{x}_B|^{3/2}}{\sqrt{2G(M_A + M_B)}}. \quad (1)$$

This slightly underestime that infall timescale, because the galaxies do not start at rest. It nevertheless give a timescale which is sufficient for our purposes (which is to select the time of visualisation outputs).

Initialisation of stellar disk

We will now extend the model to include a very simple stellar disk around galaxy A. The stellar disk will consist of N test-particles distributed on circular orbits with radii in the range, $1 \leq R \leq 5$.

With Python the disk can be initialised with the following piece of code

```
N = 512
Ncircles = 5
Theta = numpy.linspace(0.0,Ncircles*2*numpy.pi, N,endpoint=False)
R = numpy.linspace(1,5.0,N)
x = R * numpy.cos(Theta) + X0
y = R * numpy.sin(Theta) + Y0
z = numpy.zeros(N) + Z0
vtheta =  - numpy.sqrt(G*M0/R)#prograde
#vtheta = numpy.sqrt(G*M0/R)#retro-grade
vr = 0.0
vx = vr * numpy.cos(Theta) - numpy.sin(Theta) * vtheta + VX0
vy = vr * numpy.sin(Theta) + numpy.cos(Theta) * vtheta  + VY0
vz = numpy.zeros(N) + VZ0
```

The sign of v_θ can be change to make the merger prograde or retrograde.

I have implemented the above model in **BridgesAndTails_PartII.py**. The program can be modified to study different mass ratios (change the M1 parameter), different impact parameters (change X1) and the orientation of the stellar disk (change the line defining v_θ).

- Run the code to visualise the formation of bridge and tail in the default setup.
- Vary `X1`, so the impact parameter gets smaller. How does it affect the properties of the bridge and the tail?
- Change `M1` to detremine the smallest merger mass ratio, where a bridge and a tail is formed. In each case adjust X1, so we get a minimum distance (which is written in the plotting window) of $5 - 10$ in the length units.
- What happens if the disk is counter-rotating instead of co-rotating?

For more inspiration, see Toomre & Toomre (1972) and Privon et al. (2013).

References

Privon G. C., et al., 2013, ApJ, 771, 120

Toomre A., Toomre J., 1972, ApJ, 178, 623