

# Chapter 11

## Animation of Planetary Motion

### 11.1 Introduction

In lecture, we applied the Verlet method to simulate planetary motion in the presence of massive sun, and produced an animation showing the trajectory of the planet. There is no assignment in this chapter. This is just to record the outcome of our exercise, for fun.

To use animation in our notebook, we cannot use the inline option for plotting, so we start things off in the notebook style instead:

```
%pylab notebook
```

We worked through a simple example of an animated function:

```
from matplotlib.animation import FuncAnimation

x = np.linspace(0, 2 * np.pi, 100)
ya = np.cos(x)
yb = np.sin(x)

plt.xlim(0, 2 * np.pi)
plt.ylim(-1.5, 1.5)

la, = plt.plot([], [], "b-")
lb, = plt.plot([], [], "r--")

def animate(i):
    la.set_data(x[:i], ya[:i])
    lb.set_data(x[:i], yb[:i])

anim = FuncAnimation(plt.gcf(), animate, frames=100, interval=10, repeat=True)
plt.show()
```

Notice the use of the `set_data` function to update the plotted figures `la` and `lb`. You have a lot of options for how to handle this.

Next we implemented the Verlet method for a two-d system as:

```
def verlet(tau, xa, xb, ya, yb, ax, ay):
    txa = xa
    tya = ya
    xa = 2 * xa - xb + tau ** 2 * ax
    ya = 2 * ya - yb + tau ** 2 * ay
    return xa, txa, ya, tya
```

And simulated the planetary motion as:

```
vz = 1.0
tau = 0.01
xa = 1
xb = 1
ya = 0
yb = -vz*tau
az = 1
tx = np.array([])
ty = np.array([])

for i in range(int(100/tau)):
    tx = np.append(tx, xa)
    ty = np.append(ty, ya)
    ax = -az*xa/np.sqrt(xa**2+ya**2)**3
    ay = -az*ya/np.sqrt(xa**2+ya**2)**3
    xa,xb,ya,yb = verlet(tau,xa,xb,ya,yb,ax,ay)
plt.xlim(-2,2)
plt.ylim(-2,2)
plt.plot(tx,ty)
```

With the trajectories saved as the arrays tx and ty, animation was produced like this:

```
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
import numpy as np
n = np.size(tx)
print(n)
plt.xlim(-2,2)
plt.ylim(-2,2)
plt.plot(0,0,"r+")
la, = plt.plot([], [], "bo")

def animate(i):
    la.set_data(tx[i],ty[i])

anim = FuncAnimation(plt.gcf(), animate, frames=n, interval=1, repeat=True)
plt.show()
```