
Week 11 — *Generic Particle Method Program*
Kepler orbits : Trajectory Optimization

The goal of the present exercise is to use our particle method program to simulate the motion of planets of our solar system.

Exercise 1: *Units of the code*

1. In the file *init.csv* is stored the state of the planets at the first of January 2000. The coherent set of units is the following:
 - distance: AU= 149597870.700 km
 - time: 1 day
 - mass: $m_e = 5.97219e24$ kg where m_e is the mass of earth.
2. Use the starting point of the trajectory for the particle program and use it to simulate 365 days with a timestep of 1 day.
3. In the directory *trajectories* are stored files such that each file (one per step) contains the information of all the planets (named *step-XXXX.csv*). These trajectories are "measures" starting at the first of January 2000.
4. Verify that the dynamics correspond to what you expect but for Mercury. Indeed, an error in the calculation of the velocity has been made for Mercury so that the amplitude of this velocity is wrong (but not its direction). What follows aims at correcting this initial velocity.

Exercise 2: *Compute the error of the simulation*

The goal of the present exercise is to calculate the integral error for

$$E(p) = \sqrt{\sum_{i=0}^{365} | \mathbf{X}_i^{ref,p} - \mathbf{X}_i^p |^2} \quad (1)$$

where p is a given planet, $\mathbf{X}_i^{ref,p}$ is the reference position of planet p at time i and \mathbf{X}_i^p is the computed position of planet p at time i .

- Make a python function that reads the trajectory of a given planet and make a numpy array out of it.

```
def readPositions(planet_name,directory):
```

The numpy to be returned should be a $m \times n$ matrix with $m = 365$ and $n = 3$, the columns being the three components of the planet position.

- Make function that compute the above mentioned error out of two numpy trajectories.

```
def computeError(positions,positions_ref)
```

Test that function on the trajectory of Mercury

Exercise 3: *Launching the particle code from python by generating the input*

- Make a python function that generates an input file from a given input file but by scaling velocity of a given planet

```
def generateInput(scale,planet_name,input_filename,output_filename)
```

- Make a function that launches the particle code on an provided input

```
def launchParticles(exe,input,nb_steps,freq)
```

- By using all the previously mentioned functions make a function that gives the error for a given scaling velocity factor of a given planet

```
def runAndComputeError(scale,planet_name,input,exe,nb_steps,freq)
```

Exercise 4: *Optimization*

- Use all the previously defined functions and the routine

```
scipy.optimize.fmin
```

to find the correct initial velocity for Mercury.