PHY 407, Lab 1

Charlie Hughes, 1005231018

(questions 1 and 3)

Teophile Lemay, 1005036435

(questions 1 and 2)

**QUESTION 1:**

Code found in lab01\_Q1.py

b)

Euler-Cromer equations with the terms for x and y acceleration taken from the Newtonian orbit equations.

PSEUDOCODE:

# define constants (G, AU)

# define

# define Kg

# define Kg

# define . Define

# create arrays for time, position and velocity during orbit

# Run “for” loop for set period of time

# calculate using eq 3. Calculate using eq 4

# calculate using and eq 1. Calculate using and eq 2

# Plot vs time, vs time, vs time, and vs time

# Plot vs. to describe orbit

c)

Shape

Description automatically generated with low confidenceChart, histogram

Description automatically generatedChart, line chart, histogram

Description automatically generatedFigures: Plots describing Mercury’s orbit: x and y velocity with respect to time, and x vs. y position to draw out the orbit. The planetary motion was calculated over 1 earth year, assuming a purely Newtonian orbit.

d)

Gravitational forces are not actually purely Newtonian, and their effect on planetary orbits changes when considering general relativity. The figure below shows a plot of Mercury’s orbit over 4 earth years, calculated using an exaggerated correction term for general relativity. As shown in the plot, general relativity induces a precession of Mercury’s orbit. However, the actual precession of Mercury’s orbit due to general relativity would not be as dramatic as is described by the plot.

Diagram

Description automatically generatedFigure: Plot of Mercury’s orbit with exaggerated general relativity correctional term in the x, y plane

**QUESTION 2:**

Code found in lab01\_Q2.py

Planetary orbits depend not only on gravitational force of the sun, but also on the forces exerted by other planets in the system. Here, we model the effect of Jupiter on the orbit of the earth.

a)

new equations of motion:

We can continue to use equations 1 and 2 for position, but we must also consider the force that Jupiter applies to the earth’s orbit

In order to quantify Jupiter’s effect, Jupiter’s orbit is first calculated so that it can be referred to when calculating the motions of the earth.

PSEUDOCODE:

# Define constants (G, Msun, Mjuptiter, Mearth, AU, )

# set initial conditions for Jupiter x, y position; Jupiter x, y velocity; earth x, y position; earth x, y velocity

# Calculate Jupiter’s orbit (effect of earth on Jupiter considered negligible)

# create arrays to hold values for time; x, y position; x, y velocity

# Run “for” loop over time array:

# calculate using eq 3. Calculate using eq 4

# calculate with and eq 1. Calculate with and eq 2.

# calculate earth’s orbit, using the previously calculated values for Jupiter’s position

#create arrays to hold values for x, y position; x, y velocity

# run “for” loop over time array:

# calculate using eq 5. Calculate using eq 6

# calculate with and eq 1. Calculate with and eq 2.

# plot all orbits (x vs. y)

Diagram

Description automatically generated with low confidence

Figure: Plot of Jupiter and the earth’s orbits over 10 years in the x, y plane.

As shown in the plot, Jupiter does not have a dramatic effect on the earth’s orbit around the sun.

b)

Chart

Description automatically generatedIn order to better understand the effect of other planets, we increase Jupiter’s mass by a factor of 1000, and calculate the effect on the earth’s orbit.

Figure: Plot of Jupiter and the earth’s orbit in the x, y plane with Jupiter’s mass increase x1000 i

If the simulation is carried out for a longer period of time, the eccentricity of the earth’s orbit increases until the earth is thrown away for the sun, moving with constant, non-zero x and y velocity. Thus, we can conclude that Jupiter does have an effect on the earth’s orbit, albeit a small one due to the Jupiter’s relatively small mass compared to the sun and the large distance between the earth and Jupiter.

c)

**Diagram

Description automatically generated**Since gravitational force is inversely proportional to distance, an object closer to Jupiter would experience a greater perturbation to its orbit around the sun. By replacing the earth with an asteroid, further out from the sun, and calculating its orbit while accounting for Jupiter, we see greater perturbation to the orbit of the asteroid than was observed for earth.

Figure: Plot of Jupiter and an asteroid’s orbits around the sun in the x, y plane**QUESTION 3**

Code found in lab01\_Q3.py

3 a)

# define N = random samples

# define M = number of bins in histogram

# define array **Y**, filled with zeros and of size M

# find minimum & maximum of dataset N

# Define linspace array **X** using the min/max found above as the bounds and number of elements = M

# For loop ***i*** through range() array of size of M-1:

# **Y**[***i*** + 1] = size(number of elements in N >= **X[*i*]** and N < **X[*i+1*]**)

# Plot step graph of (X, Y) filling in under the graph (to make it look like a histogram)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Number of Samples | 10 | 102 | 103 | 104 | 105 | 106 |
| Time (s) | 0.048 | 0.047 | 0.047 | 0.055 | 0.146 | 3.07 |

Table 1: Time to execute custom histogram function:

b)

Chart, line chart

Description automatically generated

Figure: Averaged Runtime of Histogram Functions vs Sample Size on log scale

Evidently, the custom histogram function performs effectively as well as the function provided by *NumPy* when the number of samples ≤ 105. For large values, *NumPy* performs far better (remaining consistently around 0.5 seconds). While some of this may be attributable to optimization on the NumPy code (such as using a more efficient form than a ‘for’ loop to go through each bin), it is also likely due to NumPy working on the C level – using Python lists will require more CPU and memory overhead.

Notably, the time to execute the histogram code decreased slightly as the number of samples were increased until 104. This could be attributed to a variety of factors such as: the computer caching the code for faster execution each subsequent time, or the CPU increasing its frequency (and therefore processing power) in response to the greater load.