Simulate Planetary Trajectories

PHY 480-001 Project

Minrui Wei

Yuhan Zhu

Introduction

Questions we want to solve for our projects:

- Simulate planetary trajectories by the Euler Method
- Simulate planetary trajectories by Runge-Kutta Method
- Make an animation for planetary trajectories

Physics Principle

$$\vec{F} = -\frac{GMm}{r^3}\vec{r}$$

- The law of planetary motion is Newton's second law of motion:
  - A two-dimensional motion.
  - Two coordinate variables (x, y) and two velocity components (vx, vy)
  - Two acceleration components (ax,ay).
- To simplify the formulation we choose an appropriate unit system:
  - Kepler's 3rd law
  - T= 1 year
  - r = 1 AU
  - GMs= 4π^2
- The initial value of the earth's motion in this unit :
  - r = (1,0)
  - v = (0,2π)

$$\frac{r^3}{T^2} = \frac{GM_s}{4\pi^2}$$

$$\frac{d\vec{r}}{dt} = \vec{v}$$

$$\frac{d\vec{v}}{dt} = \vec{a} = \frac{\vec{F}}{m} = -\frac{GM_s}{r^3}\vec{r}$$

Method-Euler Method & Semi-implicit Euler method

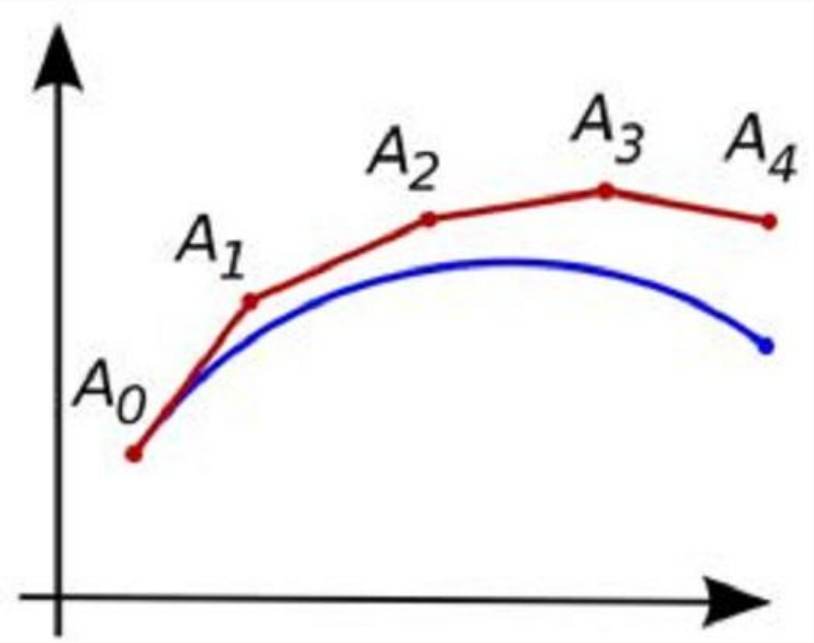
- Euler's method:
  - a first-order numerical method for solving ODE.
  - $\vec{r}(t+dt) \cong \vec{r}(t) + \vec{v}(t)dt$

    $\vec{v}(t+dt) \cong \vec{v}(t) + \vec{a}(t)dt$

- Semi-implicit Euler method:
  - An improved Euler method.
  - Update the velocity first, then use the updated velocity to update the position.

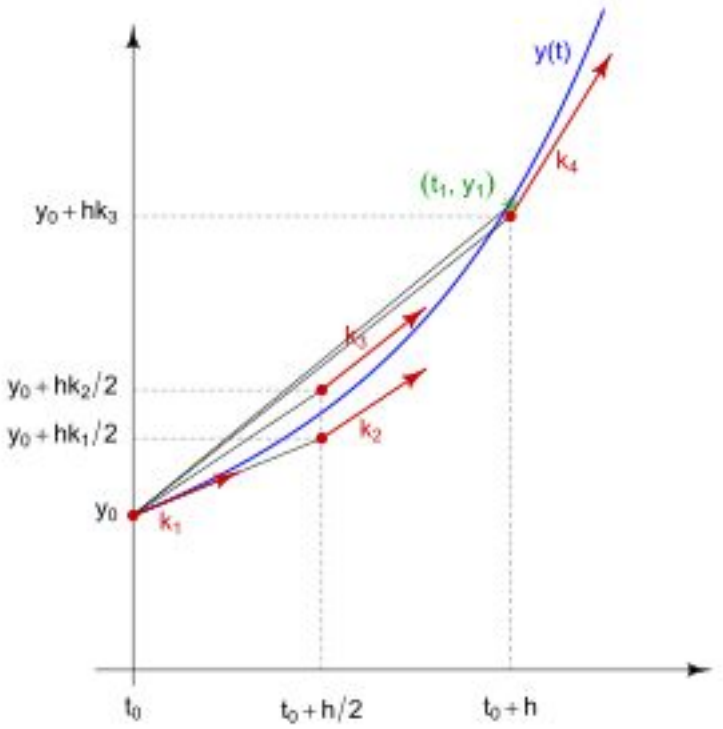
    $\vec{v}(t+dt) \cong \vec{v}(t) + \vec{a}(t)dt$

    $\vec{r}(t+dt) \cong \vec{r}(t) + \vec{v}(t)dt$



Runge-Kutta 4th order method

- The Runge-Kutta 4th order method has high precision, and measures are taken to suppress the error.
- This method uses computational program when the derivative and initial value of the equation are known, eliminating the complicated process of solving the differential equation.



$$\mathbf{k}_1 = h\mathbf{f}(\mathbf{r}, t)$$

$$\mathbf{k}_2 = h\mathbf{f}\left(\mathbf{r} + \frac{1}{2}\mathbf{k}_1, t + \frac{1}{2}h\right)$$

$$\mathbf{k}_3 = h\mathbf{f}\left(\mathbf{r} + \frac{1}{2}\mathbf{k}_2, t + \frac{1}{2}h\right)$$

$$\mathbf{k}_4 = h\mathbf{f}\left(\mathbf{r} + \frac{1}{2}\mathbf{k}_3, t + h\right)$$

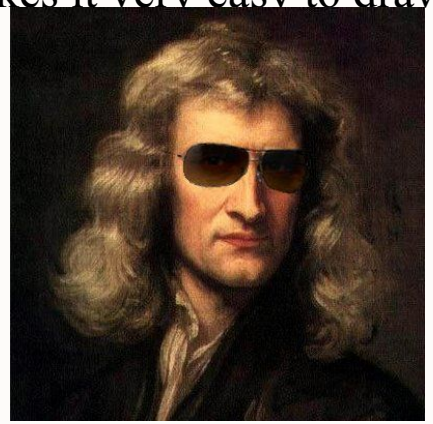
$$\mathbf{r}(t + h) = \mathbf{r}(t) + \tfrac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

Animation

Packages:

- There are many packages in Python that can be used for animation such as FuncAnimation or ArtistAnimation in Matplotlib. Or Plotly, which can be used to make nice web-based visualization

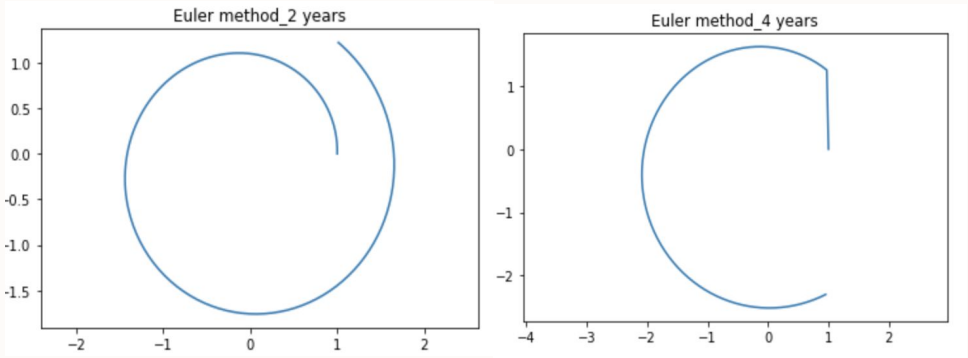
- But I ended up choosing turtle, It is extremely simple and turtle makes it very easy to draw on the screen.

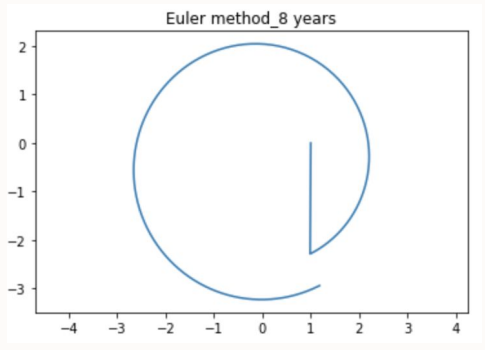


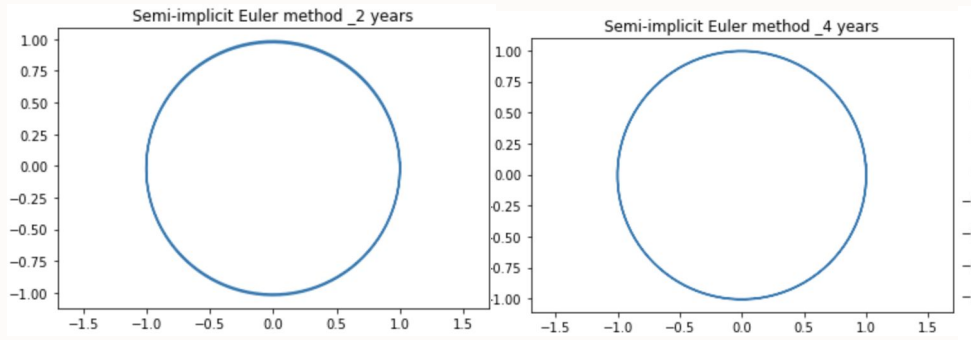
“Nature is pleased with simplicity”

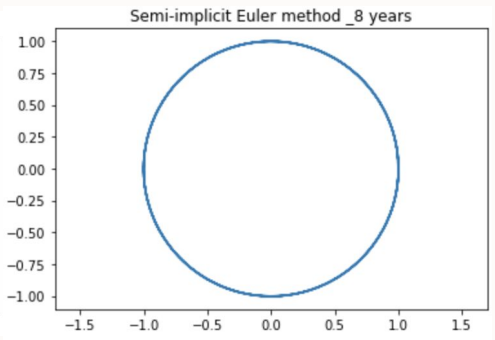
— Isaac Newton

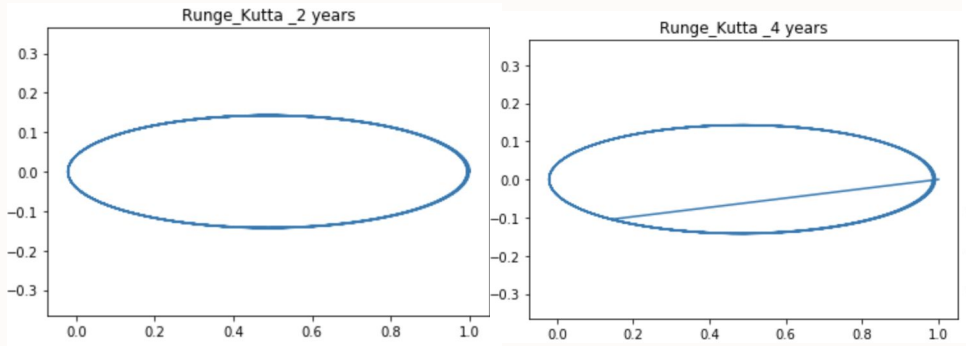
Result & Discussion

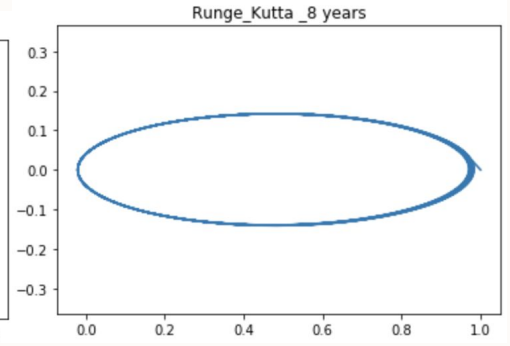




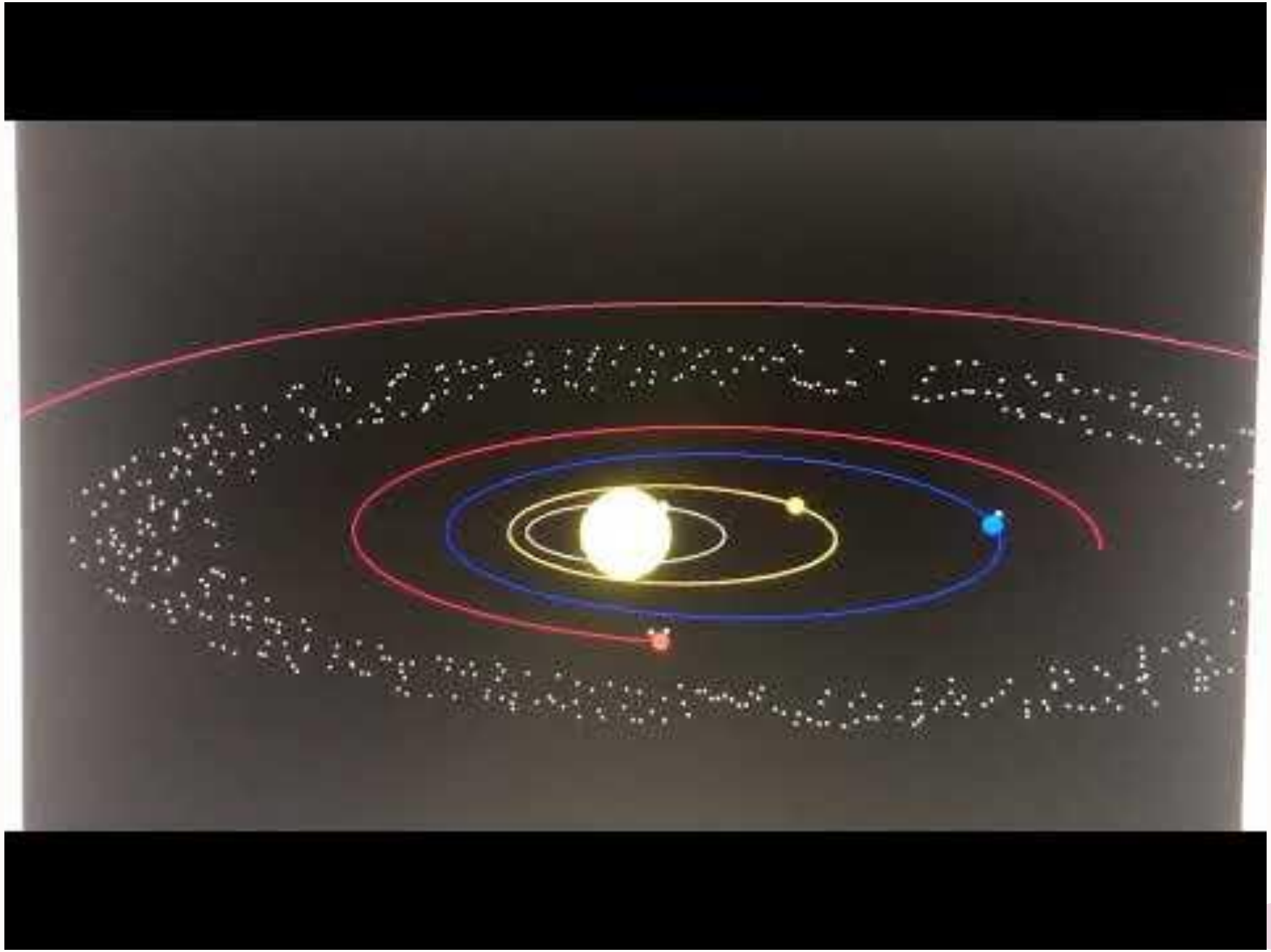






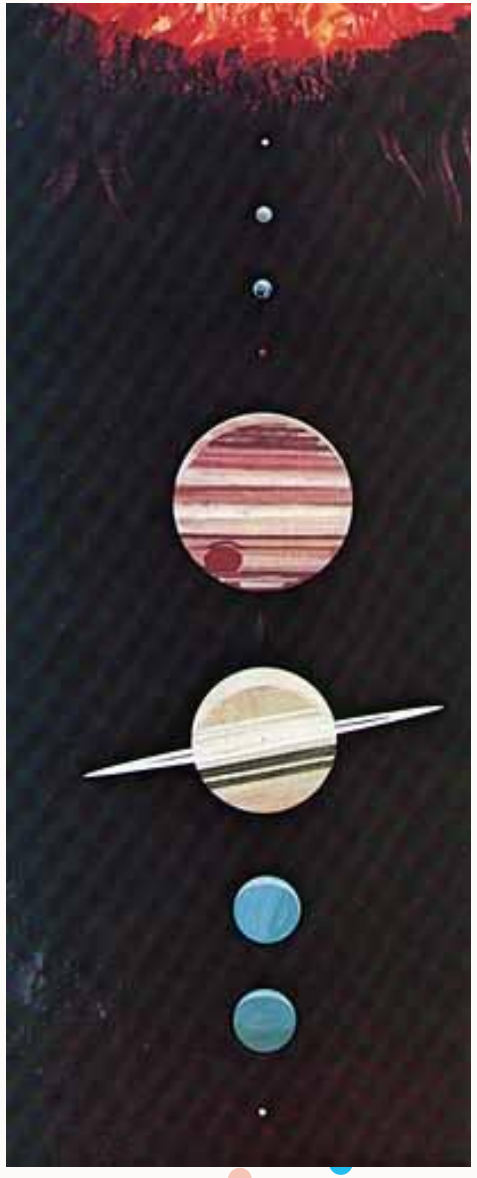


- Euler's method takes the endpoint of the tangent as the starting point for the next step. When the number of steps increases, the error will become larger and larger due to accumulation.
- Semi-implicit Euler method makes great progress in improving the accuracy of Euler method. The trajectory appears circular and show no errors as time increases
- Runge-Kutta should have the smallest error among the three methods, but we failed to simulate accurate planetary trajectory with Runge-kutta.



Discussion

- This animation simulation is very simple and easy to adjust the parameters, we can simply add or remove planets and their moons or even changing their colors.
- At the same time, the model is also very efficient. It only took about 3 seconds to generate the animation as shown in the previous slide.
- The model can easily be used to show the motion of the planets.
- The downside of this model is that we ignore the interaction between the planets and simplify the calculation of the orbital position.



Conclusion & Outlook

- Euler method and Semi-implicit, Runge-Kutta 4th order method:
  - With the number of steps increases, the error become larger for Euler method
  - Semi-implicit Euler method simulate the trajectories very well
  - Runge-Kutta 4th order method shows a large error in our result. The code must be wrong

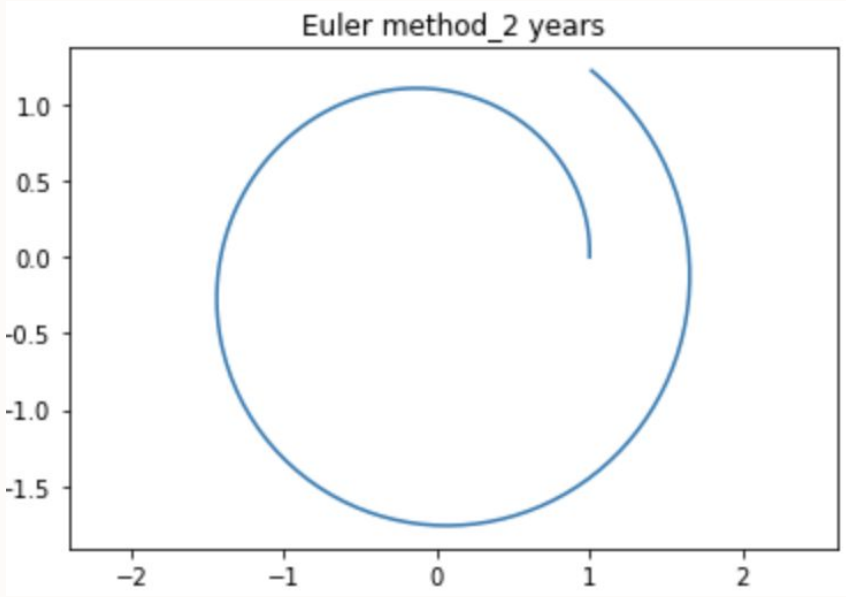
- Animation:
  - We made a good animation with Turtle.
  - We failed to apply Semi-implicit Euler method or Runge-Kutta 4th order method as a base algorithm to animation.
- In the future:
  - We fix the error of Runge-Kutta 4th order method.
  - We will apply the Semi-implicit Euler method & Runge-Kutta 4th order method as a base algorithm.
  - We will add more plants to simulate the solar system.

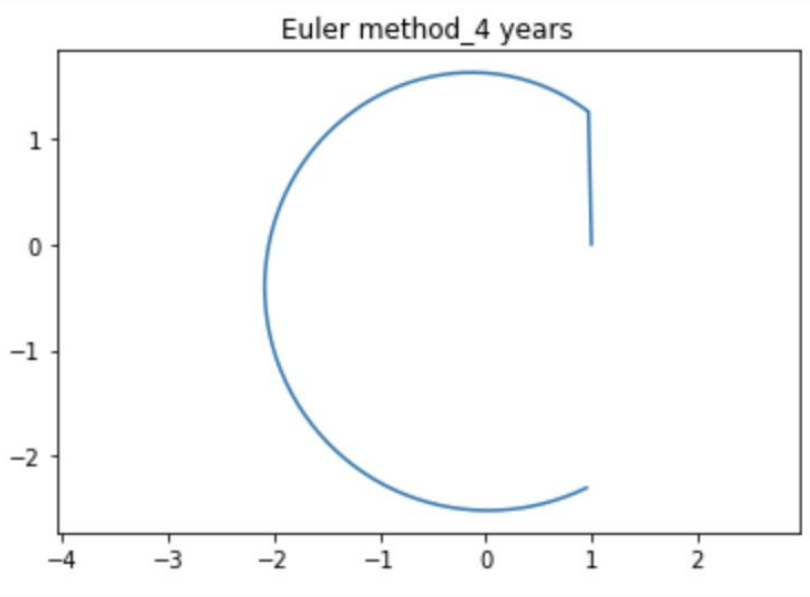
Reference

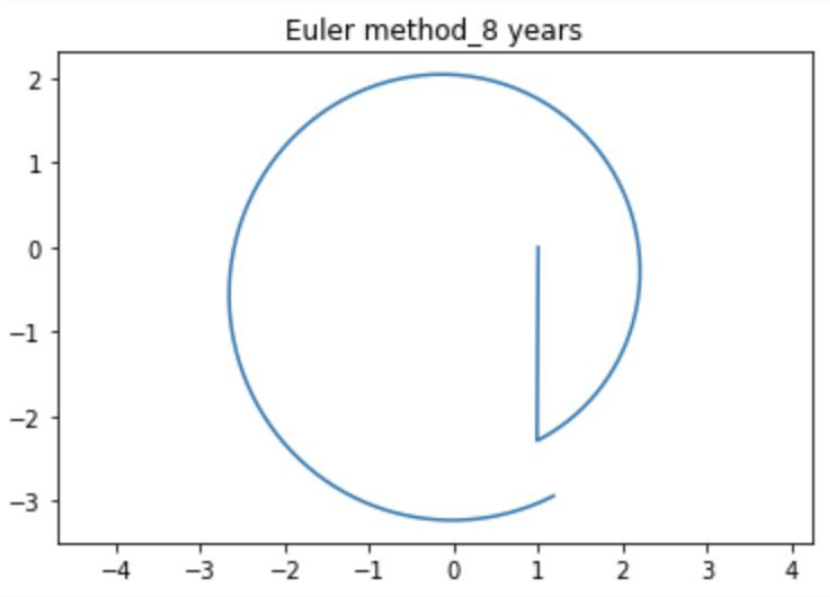
- [1] Physexp.thu.edu.tw. 2022. Nature:Physics. [online] Available at: https://physexp.thu.edu.tw/~AP/YC/PS/HTML/NUM-planetary-coding-lec.html [Accessed 22 April 2022].
- [2] Physexp.thu.edu.tw. 2022. Runge-Kutta 4th order method. Available at: https://physexp.thu.edu.tw/~AP/YC/NUM/HTML/NUM-planetary-rk4-lec.html [Accessed 22 April 2022].
- [3] GeeksforGeeks. 2022. Runge-Kutta 4th Order Method to Solve Differential Equation - GeeksforGeeks. [online] Available at: https://www.geeksforgeeks.org/runge-kutta-4th-order-method-solve-differential-equation/ [Accessed 22 April 2022].
- [4] https://docs.python.org 2022. Turtle graphics. Available at: https://docs.python.org/3/library/turtle.html [Accessed 22 April 2022].
- [5] https://docs.python.org 2022. Classes. Available at: https://docs.python.org/3/tutorial/classes.html [Accessed 22 April 2022].
- [6] NASA.gov 2022. NASA Science SOLAR SYSTEM EXPLORATION. Available at: https://solarsystem.nasa.gov/planets [Accessed 22 April 2022].
- [7] RealPython 2022. The Beginner's Guide to Python Turtle. Available at: https://realpython.com/beginners-guide-python-turtle/ [Accessed 22 April 2022].

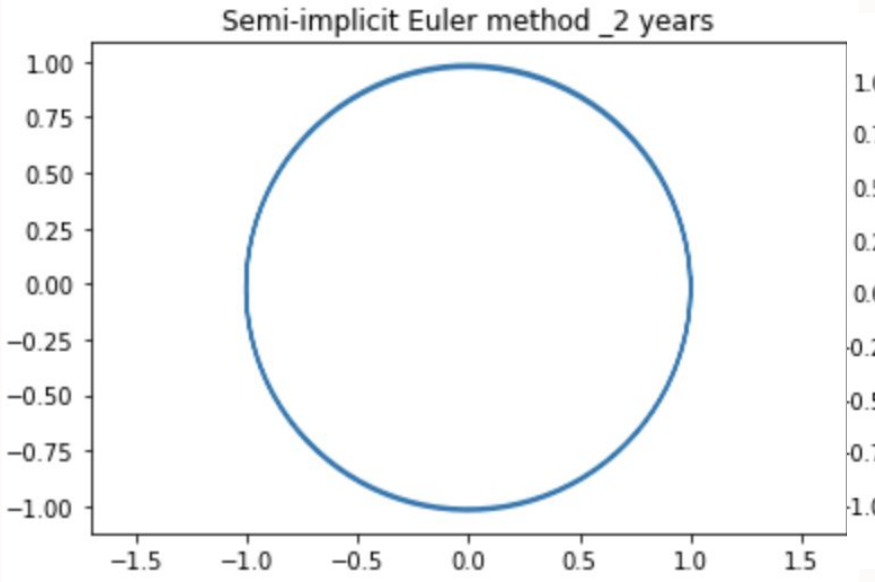
Thanks for listening！

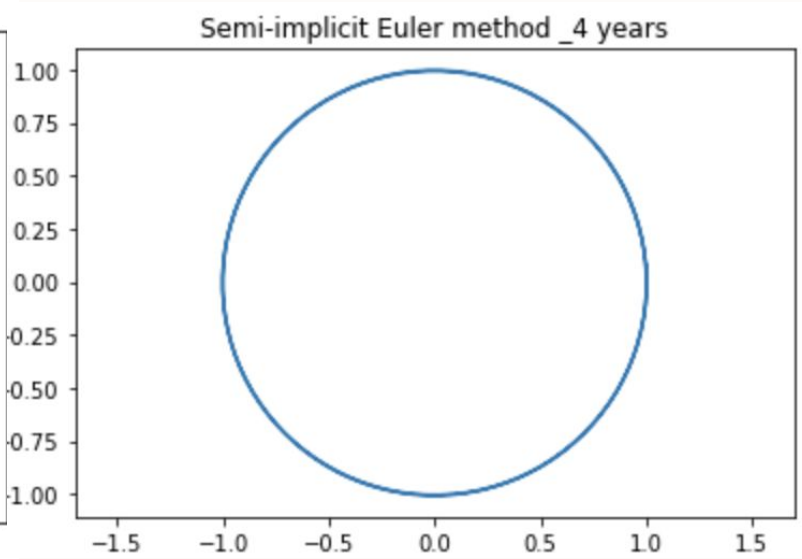
Q&A

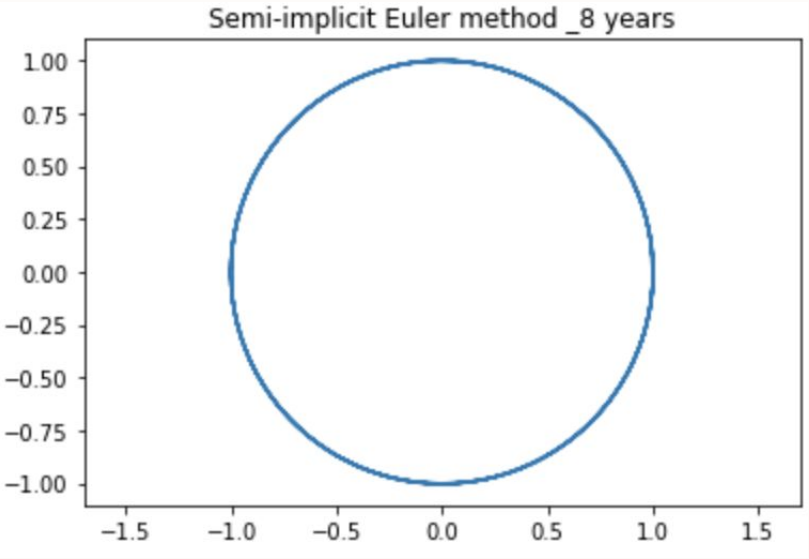

Euler method_2 years
1.0
0.5
0.0
-0.5
-1.0
-1.5
−2
−1
0
1
2

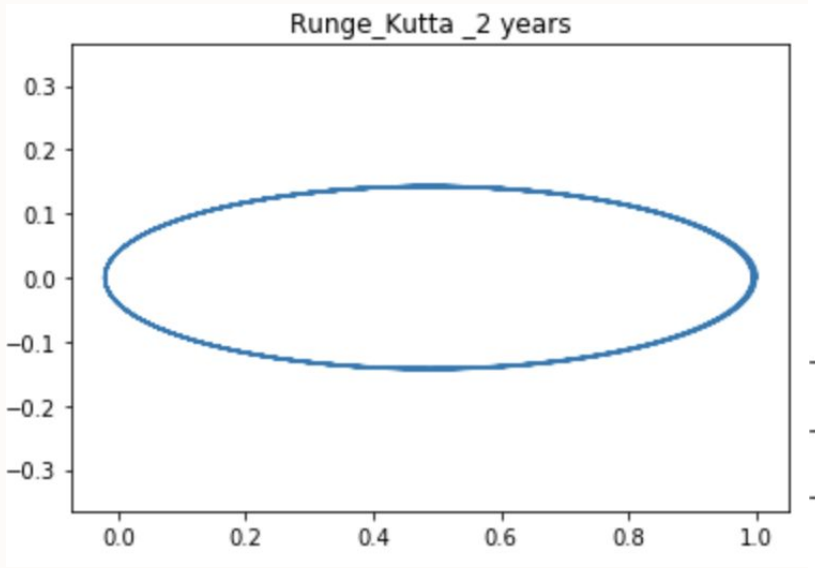

Euler method_4 years
1
0
−1
−2
−4
−3
−2
−1
0
1
2

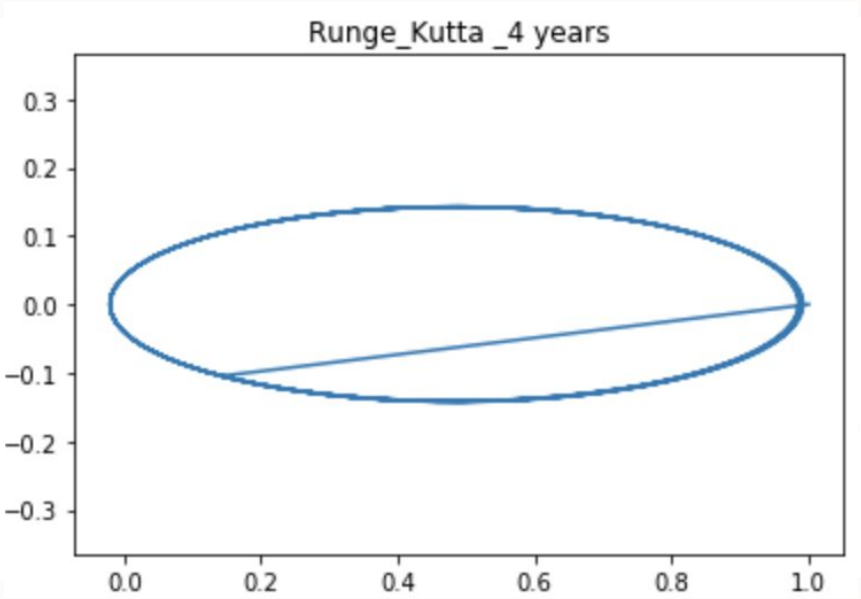

Euler method_8 years
2
1
0
−1
−2
−3
−4
−3
−2
−1
0
1
2
3
4

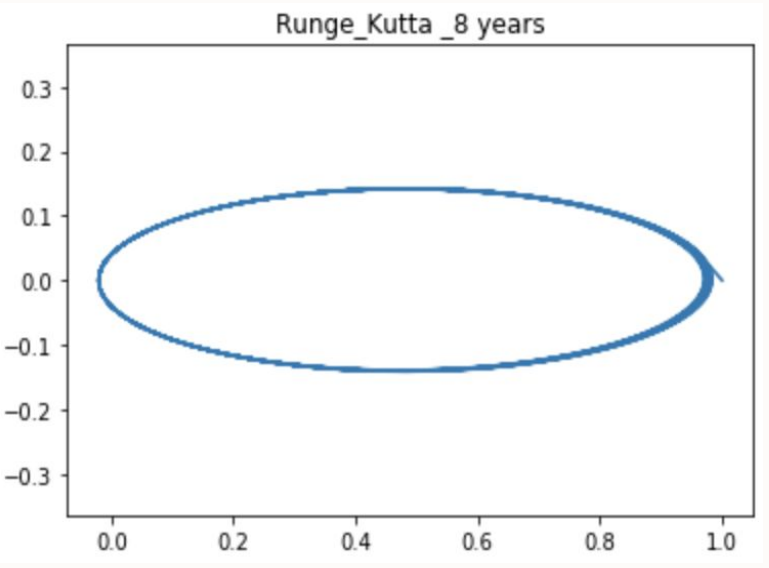

Semi-implicit Euler method _2 years
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5


Semi-implicit Euler method _4 years
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5


Semi-implicit Euler method _8 years
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5


Runge_Kutta _2 years
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
0.0
0.2
0.4
0.6
0.8
1.0


Runge_Kutta _4 years
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
0.0
0.2
0.4
0.6
0.8
1.0


Runge_Kutta _8 years
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
0.0
0.2
0.4
0.6
0.8
1.0