Солнечная система

Работу выполнила: К.С. Галиченко
Научный руководитель: А.С. Байгашов

Аннотация

Работа посвящена моделированию Солнечной системы на языке программирования Python. Полученный результат наглядно отражает размеры планет Солнечной системы, а также отношение скоростей, с которыми планеты движутся по орбитам.

Введение

Цель моего проекта заключается в применении полученных в процессе обучения математическому моделированию навыков и создание кинематической модели Солнечной системы на языке программирования Python. Солнечная система – планетная система, включающая в себя центральную звезду – Солнце – и все естественные космические объекты, вращающиеся вокруг солнца. Основной целью созданной мной модели является демонстрация движения планет относительно друг друга.
Для решения этой задачи мне предстоит смоделировать Солнце, восемь планет и Плутон, а также их орбиты, затем создать функцию, которая заставит планеты двигаться по орбитам с определёнными скоростями, зависящими от сидерического периода планет.

Постановка задачи

Для осуществления движения планет по орбитам необходимо создать две функции. Первая задаёт движение планеты по орбите, вычисляя угловую скорость планеты (ω) в зависимости от её сидерического периода (T):

$$\boldsymbol{\omega = \frac{2\pi}{T}}$$

Орбиты планет в моей работе являются эллипсами, поэтому координаты объектов x и y задаются уравнением эллипса:

$$\begin{cases} \boldsymbol{x = a \cdot cos\,\alpha} \\ \boldsymbol{y = b \cdot sin\,\alpha'} \end{cases} \quad \text{где} \quad \boldsymbol{\alpha = \omega \cdot t},$$

a и b – большая и малая полуось эллипса (a > b)

В моей работе реальными являются сидерические периоды планет, а их траектории-эллипсы заданы так, будто мы смотрим на них под некоторым углом. Это сделано, потому что реальные орбиты располагаются довольно далеко друг от друга и для красоты значения большой и малой полуоси взяты неправильно. Эллипсы задаются уравнением:

$$\begin{cases} \boldsymbol{x = a \cdot cos\,\alpha} \\ \boldsymbol{y = b \cdot sin\,\alpha'} \end{cases} \quad \text{где } \boldsymbol{\alpha \in (0;\ 3\pi)},$$

a и b – большая и малая полуось эллипса (a > b)

Начальные условия и параметры

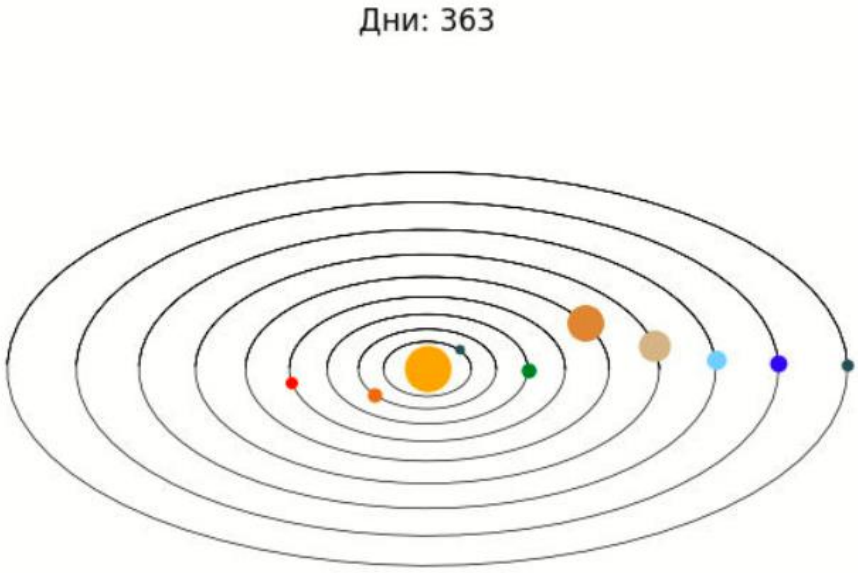
В качестве реалистичных параметров были взяты сидерические периоды планет:

Планета	Сидерический период T, сут.
Меркурий	88
Венера	225
Земля	365
Марс	687
Юпитер	4329
Сатурн	10753
Уран	30667
Нептун	60145
Плутон	90553

Для наиболее красивого графического изображения орбит планет значения a и b были подобраны в индивидуальном порядке (см. листинг кода).

Результаты моделирования

В результате численного моделирования были получены следующие результаты: динамичная модель Солнечной системы:



Заключение

Таким образом, я применила полученные методы численного моделирования, и смогла создать модель Солнечной системы, которая наглядно отражает скорости движения планет по орбитам. Результат моей работы приведён выше.

Листинг кода:

```
import matplotlib.animation as animation
import matplotlib.pyplot as plt
import numpy as np
from mpl_toolkits.axisartist.axislines import Subplot

fig = plt.figure()
ax = Subplot(fig, 111)
fig.add_subplot(ax)

ellips1, = plt.plot([], [], '-', color='black', label='ellips1', lw=0.5)
ellips2, = plt.plot([], [], '-', color='black', label='ellips2', lw=0.5)
ellips3, = plt.plot([], [], '-', color='black', label='ellips3', lw=0.5)
ellips4, = plt.plot([], [], '-', color='black', label='ellips4', lw=0.5)
ellips5, = plt.plot([], [], '-', color='black', label='ellips5', lw=0.5)
ellips6, = plt.plot([], [], '-', color='black', label='ellips6', lw=0.5)
ellips7, = plt.plot([], [], '-', color='black', label='ellips7', lw=0.5)
ellips8, = plt.plot([], [], '-', color='black', label='ellips8', lw=0.5)
ellips9, = plt.plot([], [], '-', color='black', label='ellips9', lw=0.5)
Sun, = plt.plot([], [], 'o', color='orange', label='Sun', lw=1, ms=18)
Mercury, = plt.plot([], [], 'o',color='darkslategrey',label='Mercury',lw=1,ms=3)
Venus, = plt.plot([], [], 'o', color='chocolate', label='Venus', lw=1, ms=5)
Earth, = plt.plot([], [], 'o', color='green', label='Earth', lw=1, ms=5)
Mars, = plt.plot([], [], 'o', color='red', label='Mars', lw=1, ms=4)
Jupiter, = plt.plot([], [], 'o', color='peru', label='Jupiter', lw=1, ms=14)
Saturn, = plt.plot([], [], 'o', color='tan', label='Saturn', lw=1, ms=12)
Uranus, = plt.plot([], [],'o', color='lightskyblue',label='Uranus',lw=1,ms=7)
Neptune, = plt.plot([], [], 'o', color='blue', label='Neptune', lw=1, ms=6)
Pluto, = plt.plot([], [], 'o',color='darkslategray',label='Pluto',lw=1,ms=4)

def Sun_move():
    x = 0
    y = 0
    return x, y
```

```
35  def Planet_move(T, a, b, time):
36      angle_vel = 2 * np.pi / T
37      alpha = time * angle_vel
38      x = a * np.cos(alpha)
39      y = b * np.sin(alpha)
40      return x, y
41
42  def Trajectory_move(a, b):
43      alpha = np.arange(0, 3 * np.pi, 0.1)
44      x = a * np.cos(alpha)
45      y = b * np.sin(alpha)
46      return x, y
47
48  plt.axis('equal')
49  ax.set_xlim(-350, 350)
50  ax.set_ylim(-350, 350)
51  ax.axis["right"].set_visible(False)
52  ax.axis["left"].set_visible(False)
53  ax.axis["bottom"].set_visible(False)
54  ax.axis["top"].set_visible(False)
55
56  def animate(i):
57      Sun.set_data(Sun_move())
58      Mercury.set_data(Planet_move(T=88, a=35, b=22, time=i))
59      Venus.set_data(Planet_move(T=225, a=55, b=32, time=i))
60      Earth.set_data(Planet_move(T=365, a=80, b=44, time=i))
61      Mars.set_data(Planet_move(T=687, a=110, b=58, time=i))
62      Jupiter.set_data(Planet_move(T=4329, a=145, b=74, time=i))
63      Saturn.set_data(Planet_move(T=10753, a=185, b=92, time=i))
64      Uranus.set_data(Planet_move(T=30667, a=230, b=112, time=i))
65      Neptune.set_data(Planet_move(T=60145, a=280, b=134, time=i))
66      Pluto.set_data(Planet_move(T=90553, a=335, b=158, time=i))
```

```
    ellips1.set_data(Trajectory_move(a=35, b=22))
    ellips2.set_data(Trajectory_move(a=55, b=32))
    ellips3.set_data(Trajectory_move(a=80, b=44))
    ellips4.set_data(Trajectory_move(a=110, b=58))
    ellips5.set_data(Trajectory_move(a=145, b=74))
    ellips6.set_data(Trajectory_move(a=185, b=92))
    ellips7.set_data(Trajectory_move(a=230, b=112))
    ellips8.set_data(Trajectory_move(a=280, b=134))
    ellips9.set_data(Trajectory_move(a=335, b=158))
    ax.set_title(f'Дни: {i}')


ani = animation.FuncAnimation(fig, animate, frames=365, interval=30)

plt.show()
```