

```

1  #! python3.7
2  # -*- coding: utf-8 -*-
3  from numpy import zeros, sqrt
4  from matplotlib.pyplot import style, figure, axes
5
6  # Функция f подготавливает массив, содержащий элементы вектор-функции,
7  # определяющей правую часть решаемой системы ОДУ
8  def f(u,m_sun,G) :
9      f = zeros(5)
10     # Определение исходной правой части (до перехода к длине дуги кривой)
11     f[0] = u[2]
12     f[1] = u[3]
13     f[2] = - G*m_sun*u[0]/(u[0]**2 + u[1]**2)**(3/2)
14     f[3] = - G*m_sun*u[1]/(u[0]**2 + u[1]**2)**(3/2)
15     # Добавление ещё одной компоненты вектор-функции, которая отвечает за t
16     # (выполнение формальной процедуры автономизации)
17     f[4] = 1
18     # Переопределение правой части (реализация перехода к длине дуги кривой)
19     f = f/sqrt(1 + f[0]**2 + f[1]**2 + f[2]**2 + f[3]**2)
20     return f
21
22     # Определение входных данных задачи
23     t_0 = 0.; T = 75.3*365.25*24*60*60
24     x_0 = 5.24824*10**12; y_0 = 0.
25     v_x_0 = 0.; v_y_0 = 0.9*10**3
26     G = 6.674301515151515*10**(-11)
27     m_sun = 1.98847*10**30
28
29     # Определение шага сетки вдоль интегральной кривой
30     dl = 2.*10**10
31
32     # Выделение памяти под массив сеточных значений решения системы ОДУ
33     # В строке с номером j этого массива хранятся сеточные значения решения,
34     # соответствующие j-ому узлу сетки вдоль дуги интегральной кривой
35     # Число J, определяющее число строк массива, задаём с запасом
36     J = 10000
37     u = zeros((J,5))
38
39     # Задание начальных условий
40     # (записываются в строку с номером 0 массива u)
41     u[0] = [x_0, y_0, v_x_0, v_y_0, t_0]
42
43     # Реализация схемы ERK2
44     j = 0
45     while u[j,4] < T :
46         w_1 = f(u[j],m_sun,G)
47         w_2 = f(u[j] + dl*2/3*w_1,m_sun,G)
48         u[j + 1] = u[j] + dl*(1/4*w_1 + 3/4*w_2)
49         j = j + 1
50
51     # # Реализация схемы ERK4
52     # j = 0
53     # while u[j,4] < T :
54     #     w_1 = f(u[j],m_sun,G)
55     #     w_2 = f(u[j] + dl*1/2*w_1,m_sun,G)
56     #     w_3 = f(u[j] + dl*1/2*w_2,m_sun,G)
57     #     w_4 = f(u[j] + dl*1*w_3,m_sun,G)
58     #     u[j + 1] = u[j] + dl*(1/6*w_1 + 1/3*w_2 + 1/3*w_3 + 1/6*w_4)
59     #     j = j + 1
60
61     # Отрисовка решения
62     style.use('dark_background')
63
64     fig = figure()
65     ax = axes(xlim=(-1*10**12,6*10**12), ylim=(-3.5*10**12,3.5*10**12))
66     ax.set_aspect('equal'); ax.set_xlabel('x'); ax.set_ylabel('y');
67     ax.plot(0,0,'yo',markersize=10)
68     ax.plot(u[0:j+1,0],u[0:j+1,1],'-w',markersize=5)
69     ax.plot(u[j,0],u[j,1], color='w', marker='o', markersize=7)

```

```
70 ax.set_title('Траектория движения кометы Галлея')
71
72 # Листинг программы, реализующей решение системы ОДУ
73 # после проведения процедуры перехода к длине дуги интегральной кривой
74 # (на примере моделирования движения кометы Галлея вокруг Солнца)
75 # (результатом является траектория кометы Галлея)
```