Компьютерные технологии

Горев С.

19 июня 2021 г.

Задание 8 Рассчитайте эффективную траекторию ракеты, предназначенной для наименее затратного по топливу запуска с Земли искусственного спутника Марса. Постройте зависимости скорости и координаты ракеты от времени, а также оцените точность интегрирования в зависимости от схемы интегрирования и величины шага интегрирования

1 Подход к решению задачи

Оптимальной по топливу траекторией перехода с одной орбиты на другую является Гомановская траектория - в ходе нее ракета выполняет два импульса - начальный ΔV_1, для достижения конечной орбиты, и конечный ΔV_2, для выхода на конечную орбиту. В данном решении будет использоваться Гомановская траектория полета - т.е. всего будет два импульса приращения скорости.

Теоретические значения приращения скорости

$$\Delta V_1 = V_E \left(\frac{V_E}{V_p} - 1 \right), \quad \Delta V_2 = V_M \left(1 - \frac{V_M}{V_p} \right),$$

где V_E, V_M - орбитальные скорости Земли и Марса, $V_p = \sqrt{\frac{V_E^2+V_M^2}{2}}$. Однако, данные значения справедливы только когда не учитывается гравитационное притяжение тел (Земли, Марса), поэтому при учете этих сил траектория будет искажена. В данной работе значение первого импульса будет находится путем оптимизации, условием которой будет достижение геостационарной орбиты (ГСО) Марса, при минимальной радиальной скорости ракеты при сближении.

2 Описание системы тел

В нашем случае система состоит из Солнца, Земли, Марса и ракеты. Орбиты планет будет считать круговыми, с фиксированными радиусами и орбитальными скоростями. Солнце неподвижно и расположено в начале координат.

Ракета будет запускаться с ГСО Земли, при этом будет расположена от Солнца дальше, чем Земля. Таким образом, начальные параметры ракеты можно записать как

$$x(0) = R_E + GSO_E, y(0) = 0, V_x(0) = 0, V_y(0) = V_E + V_{GSO,E} + \Delta V_1$$

, где $R_E = 1.5 \cdot 10^{11}$ м - радиус орбиты Земли, $GSO_E = 6.371 \cdot 10^6 + 35.786 \cdot 10^6 = 42.157 \cdot 10^6$ м - величина радиуса ГСО Земли, при отсчете от центра Земли, $V_{GSO,E} = 3065$ м/с - скорость на ГСО Земли.

Важным параметром является момент, в который ракета производит запуск с ГСО Земли по отношению к положению Марса. В данной работе оптимальное положение Марса будет находится путем оптимизации тем же методом, что и нахождение оптимального начального импульса.

Таким образом, в задаче будут следующие ключевые этапы:

1. Ракета начинает движение с ГСО Земли. Движение ракеты описывается всемирным законом тяготения.

2. Определяется оптимальный начальный импульс, путем минимизации некоторой функции $M = M(\Delta V_1, \theta_M(0))$, где $\theta_M(0)$ - начальный угол положения Марса в полярных координатах.

3. Найдя оптимальные значения $\Delta V_1, \theta_M(0)$, рассчитывается полет ракеты до Марса, до момента максимального сближения.

4. Дальше ракете придается дополнительный импульс ΔV_2, после которого ракета выходит на орбиту Марса.

3 Описание движения ракеты в системе

Для моделирования влияния нескольких тел на движение ракеты, воспользуемся вторым законом Ньютона и законом всемирного тяготения:

$$\vec{F} = m\vec{a} = \vec{F_g} = Gm\sum_{i=0}^{N-1}\frac{m_i}{R_i^2}\vec{e_i},$$

где i - индекс, m - масса ракеты, m_i - масса i-ого тела, R_i - расстояние между ракетой и i-м телом, $\vec{e_i}$ - единичный вектор, направленный от ракеты к телу с индексом i. Влияние ракеты на движение планет и Солнца не учитывается.

Учтем, что $\vec{r''} = \vec{a}, \vec{r'} = \vec{v}$, тогда

$$\vec{v'} = G\sum_{i=0}^{N-1}m_i\frac{\vec{r_i} - \vec{r}}{R_i^3}, \quad \vec{r'} = \vec{v}$$

где $\vec{r_i}$ - радиус вектор положения тела с индексом i. Решением полученной системы уравнений будет траектория полета ракеты $\vec{r}(t), \vec{v}(t)$. Именно эта системы и будет моделироваться в программе.

4 Определение оптимальных параметров

Ранее была введена функция $M = M(\Delta V_1, \theta_M(0))$, минимизацией значения которой будут найдены оптимальные параметры $\Delta V_1, \theta_M(0)$. Опишем критерии, по которым будет определяться нахождение оптимальных параметров.

1. Критерий достижения ГСО Марса. Минимальное расстояние до Марса должно быть меньше или равно величине ГСО Марса (считая от центра координаты Марса)

2. Минимальная радиальная скорость. Радиальная скорость ракеты при минимальном расстоянии до Марса должна быть минимальной, в идеале 0.

3. Оптимальная траектория полета. Максимальное значение радиуса орбиты ракеты должно быть близко к значению орбиты Марса.

Исходя из поставленных критериев была составлена функция, возвращаемое значение которой является

$$M = M(\Delta V_1, \theta_M(0)) = |d2M_{min}| + V_{rmin}^2 + |d2MO|,$$

где $d2M_{min}$ - расстояние до ГСО Марса при максимальном сближении, V_{rmin} - радиальная скорость при максимальном сближении, $d2MO$ - разность между максимальным радиусом орбиты ракеты и орбитой Марса. Минимизируя данную величину, будут найдены оптимальные параметры $\Delta V_1, \theta_M(0)$.

5 Результаты моделирования

Моделирование системы производится на языке Python. Используется библиотека SciPy и метод integrate.solve_ivp. В методе integrate.solve_ivp использует алгоритм Рунге-Кутты 5-го порядка ('Radau'). Шаг интегрирования выбирается так, чтобы ошибка не превышала наперед заданного значения относительной $\varepsilon_r \leq 10^{-3}$ и абсолютной ошибок $\varepsilon_a \leq 10^{-6}$ интегрирования.

Исходный код приведен в листинге 1.

Результаты моделирования приведены на рисунках 1, 2, 3.

Изначальные значения $\Delta V_1 = 2400 m/s, \theta_M(0) = 0.91 rads$. Начальные координаты и скорости ракеты

$$x(0) = 1.5 \cdot 10^{11} + 42.157 \cdot 10^6, y(0) = 0, V_x(0) = 0, V_y(0) = 29783 + 3065 + \Delta V_1$$

Оптимальные значения $\Delta V_1, \theta_M(0)$, найденные в ходе оптимизации

$$\Delta V_1^{opt} = 2327.05 m/s \quad \theta_M(0)^{opt} = 0.925 rads,$$

для сравнения, теоретической значение $\Delta V_1 = 2943.45$ м/с, однако из-за влияния гравитации Земли оптимальное значение отличается от теоретического.

Момент достижения Марса отмечен на графиках вертикальной линией, и составляет $t_{ETA} = 261.13$ дней с момента запуска. В момент достижения Марса сближение с центром Марса составляет 10475 км, при величине ГСО Марса 20389.5 км. Радиальная скорость при максимальном сближении составляет -258 м/с.

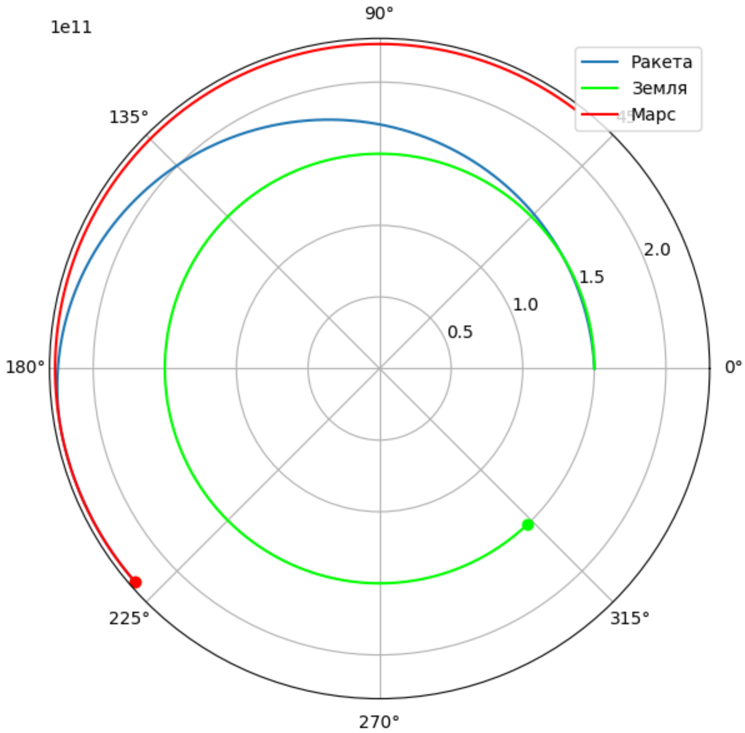

Рис. 1: Траектории движений

После достижения Марса, ракете придается второй импульс $\Delta V_2 = 2647.9$ м/с, что является теоретическим значением. После этого, ракета выходит на орбиту Марса, что можно пронаблюдать на графике скорости на рис. 3 - после момента достижения Марса, скорость начинает сильно осциллировать, что характерно при выходе на орбиту планеты. Также приведен график координаты в увеличенном мастшатбу на рис. 4, на котором наблюдается осцилляция вокруг координаты Марса.

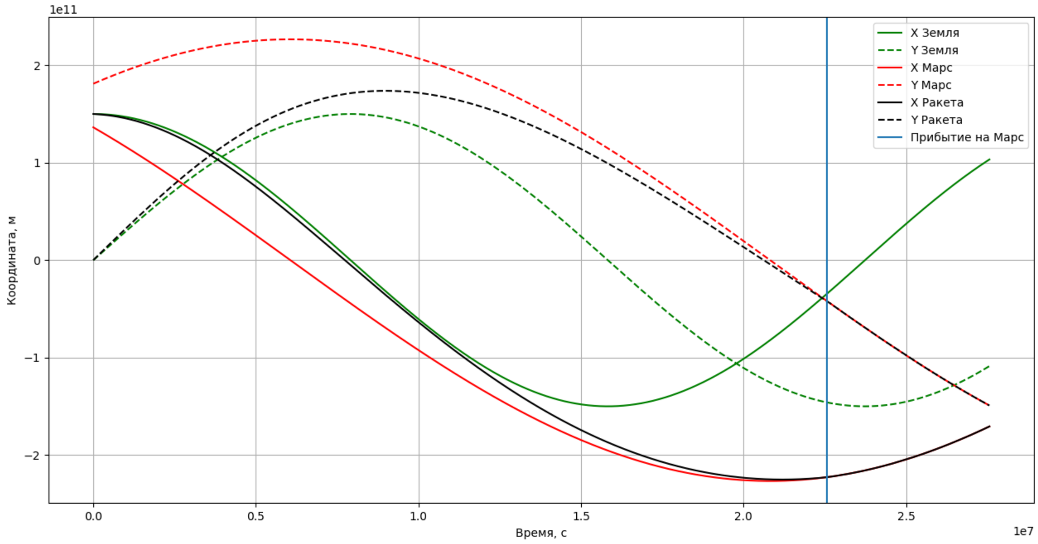

Рис. 2: Координаты от времени

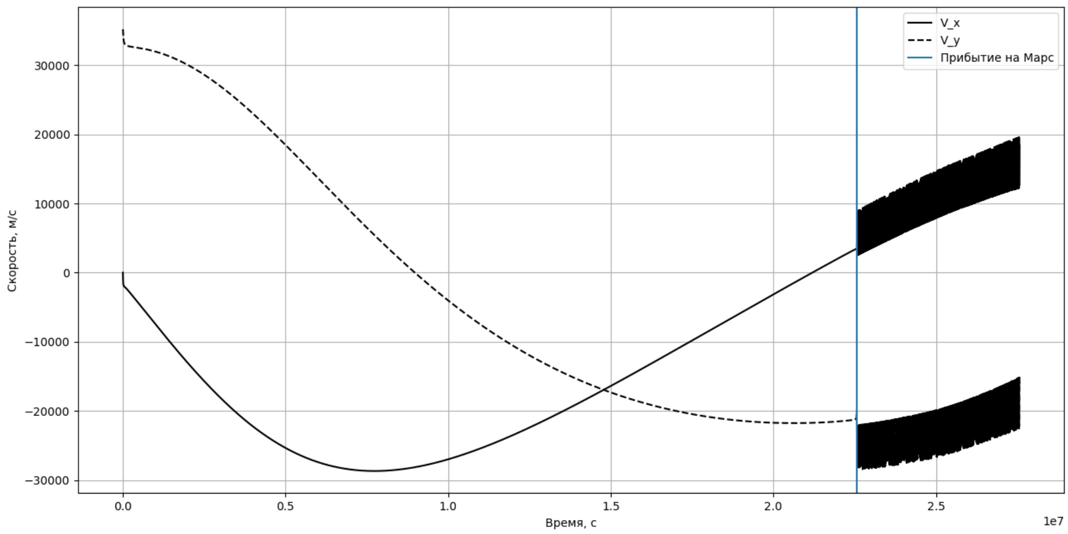

Рис. 3: Скорость ракеты от времени

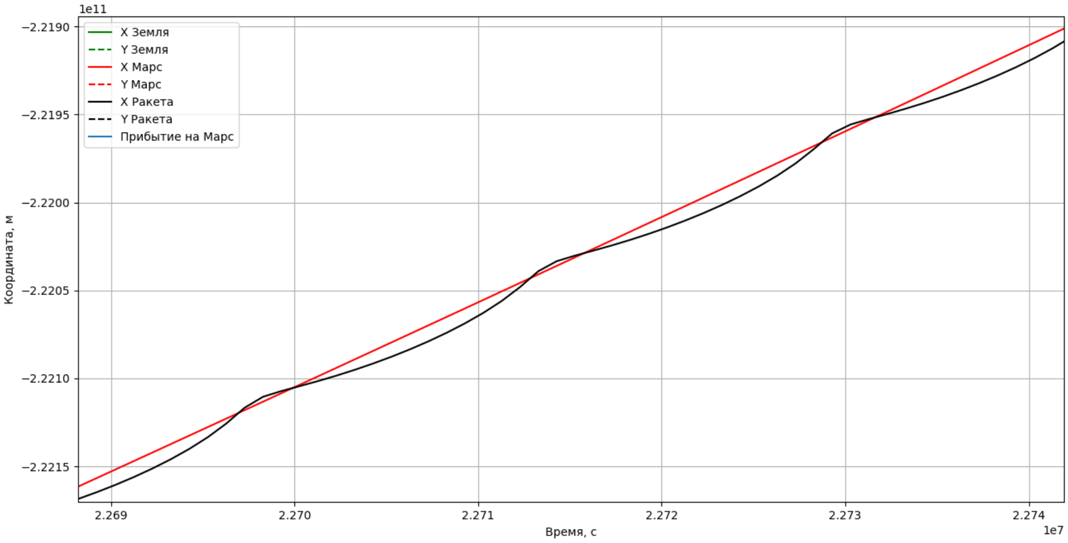

Рис. 4: Выход ракеты на орбиту Марса - координата ракеты осциллирует вокруг координаты Марса

6 Исходный код

```
# Задание 9. Рассчитайте эффективную траекторию ракеты, предназначенной для наименее
# затратного по топливу запуска с Земли искусственного спутника Марса. Постройте зависимости
# скорости и координаты ракеты от времени, а также оцените точность интегрирования в
# зависимости от схемы интегрирования и величины шага интегрирования

import numpy as np
from scipy.integrate import solve_ivp
from scipy.optimize import minimize
import matplotlib.pyplot as plt

# Константы
G = 6.67430*10**(-11)
M_SUN = 1.989*10**30  # kg

# Параметры Марса и его орбиты
M_MARS = 6.39*10**23  # кг
R_MARS = 2.26655*10**11  # Радиус орбиты, м
V_MARS = 24130  # Орбитальная скорость, мс/
ANGLE_V_MARS = V_MARS/R_MARS  # Угловая скорость, радс/
OFFSET_MARS = 0.29*np.pi
GSO_MARS = 3389500 + 17000000 # Радиус ГСО от центра, м

# Параметры Земли и ее орбиты
M_EARTH = 5.972*10**24  # кг
```

```
R_EARTH = 150*10**9  # Радиус орбиты, м
V_EARTH = 29783  # Орбитальная скорость, мс/
ANGLE_V_EARTH = V_EARTH/R_EARTH  # Угловая скорость, радс/
GSO_EARTH = 6371000 + 35.786*10**6 # Радиус ГСО от центра, м

M_ROCKET = 3*10**6 # Масса ракеты

# Теоретические величины импульсов
Vp = np.sqrt((V_MARS**2 + V_EARTH**2)/2)
dV1 = V_EARTH*(V_EARTH/Vp - 1)
dV2 = V_MARS*(1 - V_MARS/Vp)

def acc_g(t, r, offset_mars):
    ''' Рассчет суммарного ускорения от трех тел -
        Земли, Марса и Солнца.
    '''
    x, y = r
    x_E = R_EARTH*np.cos(ANGLE_V_EARTH*t)
    y_E = R_EARTH*np.sin(ANGLE_V_EARTH*t)
    x_M = R_MARS*np.cos(ANGLE_V_MARS*t + offset_mars)
    y_M = R_MARS*np.sin(ANGLE_V_MARS*t + offset_mars)

    # Расстояния до тел
    R2E = np.sqrt((x-x_E)**2 + (y-y_E)**2)
    R2M = np.sqrt((x-x_M)**2 + (y-y_M)**2)
    R2S = np.sqrt(x**2 + y**2)

    # Ускорения от каждого тела
    acc_E = M_EARTH/R2E**3 * np.array([x_E - x, y_E - y])
    acc_M = M_MARS/R2M**3 * np.array([x_M - x, y_M - y])
    acc_S = -M_SUN/R2S**3 * np.array([x, y])

    acc = G*(acc_E + acc_M + acc_S)
    return acc


def model(t, params, offset_mars):
    ''' Модель для моделирования - рассчет
        дифференциального уравнения движения
    '''
    r = params[0:2]
    v = params[2:]
    acceleration = (acc_g(t, r, offset_mars))
    return np.hstack((v, acceleration))


def find_optimal_parameters(params):
    ''' Функция для минимизации возвращаемого значения.
```

```
        Оптимальными будут те параметры, при которых
        расстояние до Марса будет меньше величины ГСО, радиальная скорость
        в максимуме Орбиты будет минимальна, а также максимальная величины
        радиуса орбиты ракеты будет максимально близка к орбите Марса.
    '''
    dv, offset_mars = params
    # Расчет траекотрии при заданных параметрах
    sol = solve_ivp(model, [0, t_end],
                    [R_EARTH + GSO_EARTH, 0, 0, V_EARTH + 3065 + dv],
                    args=(offset_mars,),
                    t_eval=time_points,
                    method='Radau'
                    )
    data, t = sol.y, sol.t
    x, y, vx, vy = data
    theta = np.arctan2(y, x)
    R = np.sqrt(x**2 + y**2)
    vr = vx*np.cos(theta) + vy*np.sin(theta)

    theta_M = ANGLE_V_MARS*t + offset_mars
    x_M = R_MARS*np.cos(theta_M)
    y_M = R_MARS*np.sin(theta_M)
    # Расстояние до Марса в каждый момент времени
    d2M = np.sqrt((x-x_M)**2 + (y-y_M)**2)

    min_d2M = np.min(d2M)
    ind_min = np.argmin(d2M)
    vr_at_min = vr[ind_min] # Радиальная скорость при максимальном сближении с Марсом

    max_R = np.max(R)
    d2MO = np.abs(max_R - R_MARS) # Расстояние от максимальной точки орбиты ракеты до орбиты Марса
    min_d2M -= GSO_MARS # Минимальное расстояние до ГСО Марса

    # Вывод значений при оптимизации
    # print('Imp:', dv, '\tM off:', offset_mars, '\tmin vr:',
        #    vr_at_min, '\tmin d2M:', min_d2M, ' m')
    return np.abs(min_d2M) + vr_at_min**2 + d2MO

# Первичное время моделирования
t_end = 3*10**7
dV_initial = 2400 # Изначальное предположение первого импульса, мс/
time_points = np.linspace(0, t_end, 10000)

# Моделирование с поиском оптимальной величины начального импульса и положения Марса
print("Ищем оптимальные параметры запуска...")
res = minimize(find_optimal_parameters, [dV_initial, OFFSET_MARS],
               options={'maxiter':200}, method='Nelder-Mead')
```

```
print(res.message)
opt_start_impulse, opt_offset_mars = res.x # Найденные оптимальные значения импульса и
    положения Марса
print("Теоретическое значение импульса:", dV1, 'мс/')
print("Изначальное значение импульса:", dV_initial, 'мс/')
print("Оптимальное значение импульса:", opt_start_impulse, 'мс/')
print("Изначальное положение Марса:", OFFSET_MARS, 'радиан')
print("Оптимальное положение Марса:", opt_offset_mars, 'радиан')

# Рассчет траектории с оптимальным значением начального импульса скорости
start_values = [R_EARTH + GSO_EARTH, 0, 0, V_EARTH + 3065 + opt_start_impulse]
sol1 = solve_ivp(model, [0, t_end], start_values, args=(opt_offset_mars,),
                t_eval=time_points, method='Radau')
data1, t1 = sol1.y, sol1.t
x1, y1, vx1, vy1 = data1
theta = np.arctan2(y1, x1)

theta_M = ANGLE_V_MARS*t1 + opt_offset_mars
x_M = R_MARS*np.cos(theta_M)
y_M = R_MARS*np.sin(theta_M)

D2M = np.sqrt((x1-x_M)**2 + (y1-y_M)**2)
min_D2M = np.min(D2M) # Минимальное расстояние до Марса
min_D2M_ind = np.argmin(D2M)
print('Расстояние до Марса при максимальном сближении:', min_D2M/1000,
    'км, Расстояние ГСО Марса', GSO_MARS/1000, 'км')
ETA_M = t1[min_D2M_ind]  # Время максимального сближения

# На данном этапе нам интересны только время и координаты до сближения
list_of_data = [t1, x1, y1, vx1, vy1, theta, theta_M, x_M, y_M]
for i in range(0, len(list_of_data)):
    list_of_data[i] = list_of_data[i][0:min_D2M_ind+1]
t1, x1, y1, vx1, vy1, theta, theta_M, x_M, y_M = list_of_data

vr1 = vx1*np.cos(theta) + vy1*np.sin(theta)
vr_at_min = vr1[-1] # Радиальная скорость при максимальном сближении
print("Время до максимального сближения:", ETA_M/86400, 'дней')
print('Радиальная скорость при максимальном сближении:', vr_at_min, 'м/с')

# Расчет продолжения траектории
print("Расчет продолжения тракетории")
print("Величина второго импульса:", dV2, "м/с")
extra_time = 0.5*10**7 # Дополнительное время для рассчета, с
time_points2 = np.linspace(ETA_M, ETA_M + extra_time, 5000)
start_values2 = [x1[-1], y1[-1], vx1[-1], vy1[-1] - dV2]
sol2 = solve_ivp(model, [ETA_M, ETA_M + extra_time], start_values2, args=(opt_offset_mars,),
                t_eval=time_points2, method='Radau')
```

```
data2, t2 = sol2.y, sol2.t
x2, y2, vx2, vy2 = data2

# Временный отсчеты
t = np.concatenate((t1, t2))

# Координаты ракеты
x = np.concatenate((x1, x2))
y = np.concatenate((y1, y2))
vx = np.concatenate((vx1, vx2))
vy = np.concatenate((vy1, vy2))
theta = np.arctan2(y, x)
R = np.sqrt(x**2 + y**2)

# Координаты Земли
theta_E = ANGLE_V_EARTH*t
R_E = R_EARTH*np.ones(len(theta_E))
x_E = R_EARTH*np.cos(theta_E)
y_E = R_EARTH*np.sin(theta_E)

# Координаты Марса
theta_M = ANGLE_V_MARS*t + opt_offset_mars
R_M = R_MARS*np.ones(len(theta_M))
x_M = R_MARS*np.cos(theta_M)
y_M = R_MARS*np.sin(theta_M)

#########    Блок отрисовки    #######
fig, ax = plt.subplots(subplot_kw={'projection': 'polar'})
fig2, ax2 = plt.subplots()
fig3, ax3 = plt.subplots()

# Траектории движения
ax.plot(theta, R, label='Ракета')
ax.plot(theta[-1], R[-1], 'o')
ax.plot(theta_E, R_E, color=[0, 1, 0], label='Земля')
ax.plot(theta_E[-1], R_E[-1], 'o', color=[0, 1, 0])
ax.plot(theta_M, R_M, color=[1, 0, 0], label='Марс')
ax.plot(theta_M[-1], R_M[-1], 'o', color=[1, 0, 0])
ax.legend()

# Координаты
ax2.plot(t, x_E, 'g-', label='X Земля')
ax2.plot(t, y_E, 'g--', label='Y Земля')
ax2.plot(t, x_M, 'r-', label='X Марс')
ax2.plot(t, y_M, 'r--', label='Y Марс')
ax2.plot(t, x, 'k-', label='X Ракета')
ax2.plot(t, y, 'k--', label='Y Ракета')
ax2.axvline(ETA_M, label='Прибытие на Марс')
```

```
ax2.set_xlabel('Время, с')
ax2.set_ylabel('Координата, м')
ax2.legend()
ax2.grid()

# Скорость
ax3.plot(t, vx, 'k-', label='V_x')
ax3.plot(t, vy, 'k--', label='V_y')
ax3.axvline(ETA_M, label='Прибытие на Марс')
ax3.set_xlabel('Время, с')
ax3.set_ylabel('Скорость, мс/')
ax3.legend()
ax3.grid()

plt.show()
```

Листинг 1: Исходный код задания