CaNoRock

Exercise Molniya$_1$ and CiRK$_2$

UiT - Norges Arktiske Universitet

July 10, 2024

Task a)

We have a that Molniya spacecraft have a highly elliptical orbit. With the following parameters ([1]):

Eccentricity $e = 0.74$	Semi-major axis $a = 26000km$	inclination $i = 63.5\,\text{deg}$
argument of perigee $\omega = 270\,\text{deg}$	Period $T = 718min$	Gravitational parameter $\mu = 3.986 \cdot 10^{14}$

Which is orbital parameters for a Molniya type of orbit with a right ascension of the ascending node $\Omega = 40\,\text{deg}$. To calculate the position (X,Y,Z) of the satellite (Molniya_1), with Earth centered reference frame. We can use our calculations for coordinate transformation from the lecture notes, from ellipse into X, Y and Z.

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = r \begin{bmatrix} cos(\Omega)cos(\omega+\theta) - sin(\Omega)cos(i)sin(\omega+\theta) \\ sin(\Omega)cos(\omega+\theta) + cos(\Omega)cos(i)sin(\omega+\theta) \\ sin(i)sin(\omega+\theta) \end{bmatrix} \tag{1}$$

Here we have two unknowns r and θ. Since here we just gone look at the orbit, we can sett $\theta = 360\,\text{deg} = 2\pi$, for the radius r we gone need to calculate the eccentric anomaly E. Which is done by the following formula.

$$\tan\frac{\theta}{2} = \tan(\frac{E}{2})(\frac{1+e}{1-e})^{1/2} \Rightarrow E = 2\tan^{-1}(\tan(\frac{\theta}{2})\sqrt{\frac{1-e}{1+e}}) \tag{2}$$

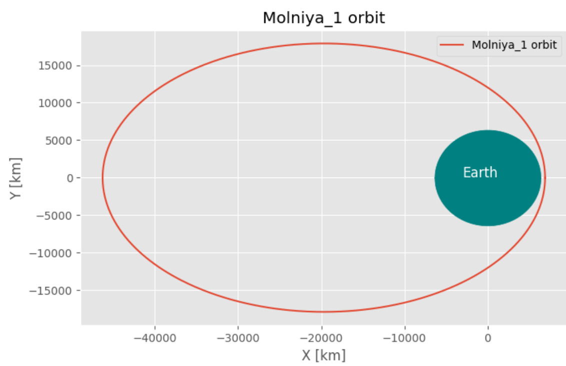
We can go forward and calculate the radius.

$$r = a(1 - e\cos E) \tag{3}$$

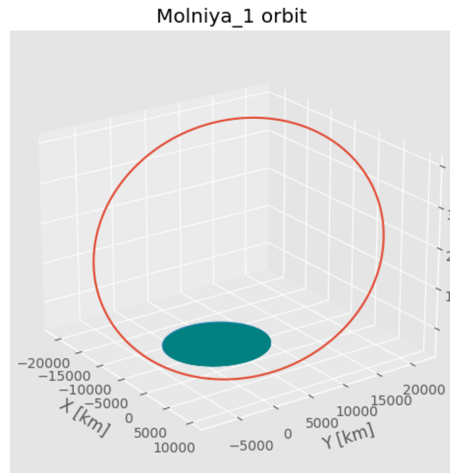
Than we put the values we get from computing the radius into 1. For plotting the orbit in two-dimensjonal, we can use the following.

$$x = r \cdot \cos(\theta) \qquad\qquad y = r \cdot \sin(\theta) \tag{4}$$

By computing all equations above, gives us the following plots for the orbit of Molniya_1.



(a) Molniya_1 2d orbit



(b) Molniya_1 3d orbit

From the plots we see that, the orbit is highly elliptical and have perigee closest to Earth.

Task b)

Now we gone assume that the satellite is passing perigee at $t_p = 0s$. To calculate how the position of the orbit changes with time, we need to execute the following. We know t, so we can find the mean anomaly a, than use Kepler's equation numerically to find E and from E we can obtain θ and r. So we start of by calculating the mean anomaly a.

$$M = \sqrt{\frac{\mu}{a^3}}(t - t_p) \tag{5}$$

Where t_p is the time of the satellite at perigee, which means that $t_p = 0$. Than we calculate E.

$$E = M + esinE \tag{6}$$

Where we use that E for the first calculation is equal to the lowest value of M, than we can iterate through the equation for the new values of E. Than use this, for the calculation of θ and r.

$$\tan\frac{\theta}{2} = \tan(\frac{E}{2})(\frac{1+e}{1-e})^{1/2} \Rightarrow \theta = 2\tan^{-1}(\sqrt{\frac{1+e}{1-e}}\tan(\frac{E}{2})) \tag{7}$$

And for the radius.

$$r = a(1 - e\cos E)$$

To show how the position is changing with time, we make one data point for every 1 hour. Which gives the following plot.

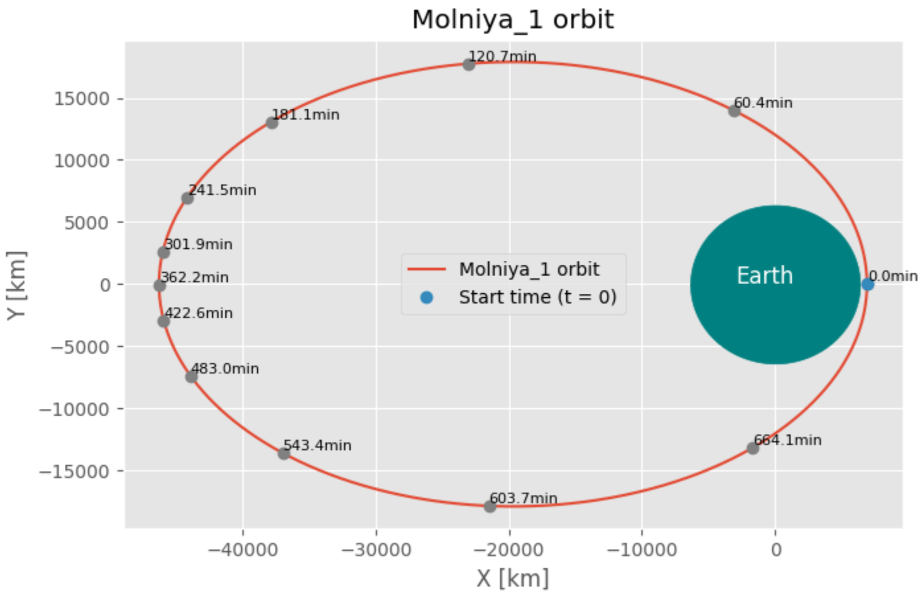


Figure 2: Position of the satellite changing with time, with one data point for every one hour. One complete orbit approx 718min.

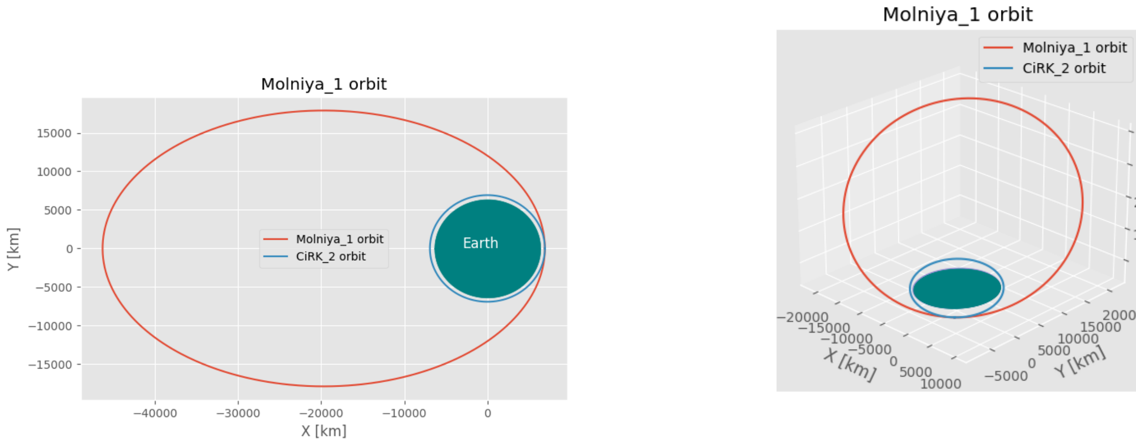
By thinking of a satellite in elliptical orbit, we would expect that at perigee it will have the highest velocity, as it goes towards the apogee the velocity will start to decrees, at apogee it will have the lowest velocity and as it goes back to perigee to will start to gain velocity again. This comes True by looking at the plot above. By thinking of the time, position and velocity argument. That would say, where the velocity is at it's highest it would spend less time than where the velocity is highest. Which we can confirm by looking at the plot and the data points that is spaced by a hour, we see that we have way more data point at apogee than at perigee. Hence, it's use more time in apogee and it's where the velocity is lowest and it's use less time in perigee where the velocity is highest.

Task c)

Now we gone plot the orbit for a new satellite called CiRK_2 that have the same parameters as Molniya_1, but the sami-major axis is equal to the radius of perigee ($a = r_p$). First we need to calculate what r_p is from the Molniya orbit, this is done by using the following formula And since we have that $a = r_p$, the eccentricity will be zero ($e = 0$).

$$r_p = a(1 - e) = a(1 - 0) = a \tag{8}$$

Running this in our program, we get that $r_p = 545km$ from Earth surface. I interpret the task, as we should use all the same parameters for CiRK_2 as we used for Molniya_1 besides changing the semi-major axis for CiRK_2. Than by plotting the orbits, and compere we get the following.



(a) Molniya_1 2d orbit (b) Molniya_1 3d orbit

We see that by having the semi-major axis (a) equal to the radius of the periabsis (r_p), and have that the eccentricity is changed to $e = 0$. The orbit becomes circular, so the satellite will maintain a relatively constant distance from the center of Earth. If we would have changed the semi-major axis and a corresponding eccentricity, we would have obtained a highly stretched out elliptical orbit. To compare the velocities between CiRK_2 and Molniya_1, we calculate their velocity in perigee and apogee. With following formula:

$$v_{perigee} = \sqrt{\mu(\frac{2}{r_p} - \frac{1}{a})} \qquad v_{apogee} = \sqrt{\mu(\frac{2}{r_a} - \frac{1}{a})} \tag{9}$$

Since CiRK_2 have a circular orbit, the velocity in perigee and apogee would be the same. So CiRK_2 have a velocity of $v \approx 7591$m/s, Molniya_1 has a velocity of $v_p \approx 10014$m/s in perigee and $v_a \approx 1496$m/s in apogee. Since the CiRK_2 have a constant velocity around the circular orbit, it will complete a full orbit way before the Molniya_1 satellite have completed a full orbit. We can compare the mean anomalies, by plotting the the mean anomalies as function of time.

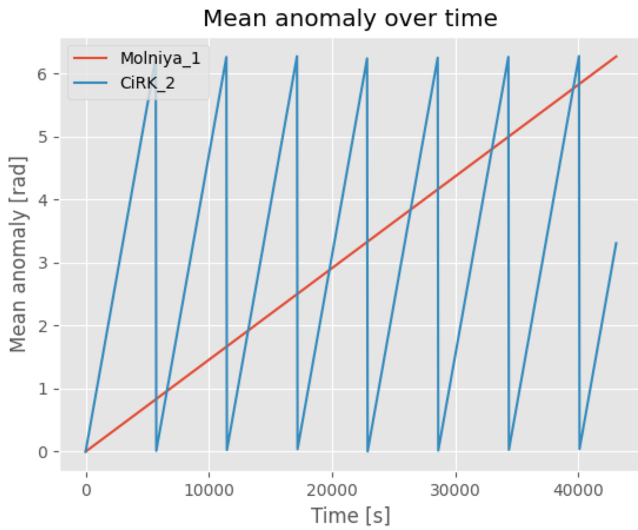


Figure 4: Comparison of the mean anomalies between the Molniya_1 and CiRK_2 satellites.

The mean anomaly is one of the six classical orbital elements used to describe the position and motion of an satellite in an orbit, the mean anomaly is an angular parameter that indicates where the satellite is in it's orbit at a given point in time. From the plot it's comes clear that the mean anomaly for CiRK_2 is varying, way more than the mean anomaly for Molniya_1. As stated above, this means that the CiRK_2 satellite is completing seven full orbits before Molniya_1 have completed one orbit. Which makes sense, by thinking of what we found for the velocity in the two different orbits.

Appendix

```python
import matplotlib.pyplot as plt
import numpy as np


plt.style.use('ggplot')


"""
-------------------------------------------------------------------
Task a)
Plot the orbit of the Molniya-1 satellite in the Earth-Centered Inertial (
    ECI) frame.
-------------------------------------------------------------------
"""

# Molniya_1 orbit parameters
inclination = np.radians(63.4) # Deg (i)
period = 718 # min
eccentricity = 0.74 # (e)
semi_major_axis = 26600e3 # km to m (a)
```

```python
OMEGA = np.radians(40) # Deg
mean_anomaly= M_0 = 0 # Deg at t = 0 (M_0)
omega = np.radians(270) # Deg (  ) argument of perigee
T = 718*60 # Period in seconds (T)

r_e = 6371e3 # Earth radius (m)
mu = 3.986e14  # Gravitational constant (m^3/s^2)  mu = G*M_earth


def task_a():
    theta = np.linspace(0, 2*np.pi, 1000) # Angle vector
    E = 2*np.arctan(np.tan(theta/2)*np.sqrt((1-eccentricity)/(1+eccentricity))) # Eccentric anomaly
    r = semi_major_axis*(1-eccentricity*np.cos(E)) # Radius

    # Calculating X, Y and Z position vectors for 3D plot
    X = r * (np.cos(OMEGA)*np.cos(omega+theta)-np.sin(OMEGA)*np.cos(inclination)*np.sin(omega+theta))
    Y = r * (np.sin(OMEGA)*np.cos(omega+theta)+np.cos(OMEGA)*np.cos(inclination)*np.sin(omega+theta))
    Z = r * (np.sin(inclination)*np.sin(omega+theta))

    # Calculating X, Y and Z position vectors for 2D plot
    X_2d = r*np.cos(theta)
    Y_2d = r*np.sin(theta)


    # Converting to km
    X = X/1000
    Y = Y/1000
    Z = Z/1000

    X_2d = X_2d/1000
    Y_2d = Y_2d/1000

    # make a circle at earths position
    earth_r = 6371 # km
    earth_theta = np.linspace(0, 2*np.pi, 1000) # Angle vector
    earth_X = earth_r * np.cos(earth_theta) # Radius
    earth_Y = earth_r * np.sin(earth_theta) # Radius
    earth_z = np.zeros(1000) # Radius

    # 2D plot of the orbit
    plt.plot(X_2d, Y_2d, label='Molniya_1 orbit')
    plt.fill_between(earth_X, earth_Y, color='teal')
    plt.text(-3000, 0, 'Earth', fontsize=12, color='white')
    plt.xlabel('X [km]')
    plt.ylabel('Y [km]')
    plt.title('Molniya_1 orbit')
    plt.legend()
    plt.show()

    # 3D plot of the orbit
    fig = plt.figure()
```

```
    ax = plt.axes(projection='3d')
    ax.plot3D(X, Y, Z)
    ax.plot3D(earth_X, earth_Y, earth_z)
    ax.add_collection3d(plt.fill_between(earth_X, earth_Y, earth_z, color='teal'))
    ax.set_xlabel('X [km]')
    ax.set_ylabel('Y [km]')
    ax.set_zlabel('Z [km]')
    plt.title('Molniya_1 orbit')
    plt.show()


"""
-----------------------------------------------------------------
Task b)
Assume that the satellite passes perigee at t=0s. Investigate and discuss
how the position of the satellite changes with time.
-----------------------------------------------------------------
"""

T = 718*60 # Period in seconds (T)
time_span = np.linspace(0, T, 1000) # Time span (s)


def task_b():
    # Position vectors
    X = np.zeros(1000)
    Y = np.zeros(1000)
    Z = np.zeros(1000)

    X_2d = np.zeros(1000)
    Y_2d = np.zeros(1000)

    for idx, t in enumerate(time_span):
        M = np.sqrt(mu/semi_major_axis**3)*t # Mean anomaly

        E = M + eccentricity*np.sin(M_0) # first calculation of E_0
        E = M + eccentricity*np.sin(E) # Eccentric anomaly Kepler's equation

        r = semi_major_axis*(1-eccentricity*np.cos(E)) # Radius
        theta = 2*np.arctan(np.sqrt((1+eccentricity)/(1-eccentricity))*np.tan(E/2)) # Angle vector

        # Calculating X, Y and Z position vectors for 3D plot
        X[idx] = r * (np.cos(OMEGA)*np.cos(omega+theta)-np.sin(OMEGA)*np.cos(inclination)*np.sin(omega+theta))
        Y[idx] = r * (np.sin(OMEGA)*np.cos(omega+theta)+np.cos(OMEGA)*np.cos(inclination)*np.sin(omega+theta))
        Z[idx] = r * (np.sin(inclination)*np.sin(omega+theta))

        # Calculating X, Y and Z position vectors for 2D plot
        X_2d[idx] = r*np.cos(theta)
        Y_2d[idx] = r*np.sin(theta)
```

```
    # Converting to km, X, Y and Z position vectors
    X = X/1000
    Y = Y/1000
    Z = Z/1000

    # Vector for plotting the orbit in 2D
    X_2d = X_2d/1000
    Y_2d = Y_2d/1000

    earth_r = 6371 # km
    earth_theta = np.linspace(0, 2*np.pi, 1000) # Angle vector
    earth_X = earth_r * np.cos(earth_theta) # Radius
    earth_Y = earth_r * np.sin(earth_theta) # Radius
    earth_z = np.zeros(1000) # Radius

    plt.plot(X_2d, Y_2d, label='Molniya_1 orbit')
    interval = 84
    for idx in range(0, 1000, interval): # Plotting the time intervals on the orbit with 60min spacing
        plt.text(X_2d[idx], Y_2d[idx], f"{(round(time_span[idx]/60, 1))} min", fontsize=8, ha='left', va='bottom')
        plt.plot(X_2d[idx], Y_2d[idx], 'o', color='gray')
    plt.plot(X_2d[0], Y_2d[0], 'o', label='Start time (t = 0)')
    plt.fill_between(earth_X, earth_Y, color='teal')
    plt.text(-3000, 0, 'Earth', fontsize=12, color='white')
    plt.title('Molniya_1 orbit')
    plt.xlabel('X [km]')
    plt.ylabel('Y [km]')
    plt.legend()
    plt.show()

    fig = plt.figure()
    ax = plt.axes(projection='3d')
    ax.plot3D(X, Y, Z)
    for idx in range(0, 1000, interval):
        ax.text(X[idx], Y[idx], Z[idx], f"{(round(time_span[idx]/60, 1))} min", fontsize=8, ha='left', va='bottom')
        ax.plot3D(X[idx], Y[idx], Z[idx], 'o', color='gray')
    ax.plot3D(X[0], Y[0], Z[0], 'o', label='Start time (t = 0)')
    ax.plot3D(earth_X, earth_Y, earth_z)
    ax.add_collection3d(plt.fill_between(earth_X, earth_Y, earth_z, color='teal'))
    ax.set_xlabel('X [km]')
    ax.set_ylabel('Y [km]')
    ax.set_zlabel('Z [km]')
    plt.title('Molniya_1 orbit')
    plt.show()


"""
-------------------------------------------------------------------
Task c)
Do the same for new satellite CiRK_2 that has the same perigee r_p as
```

```
    Molniya_1
and a semi-major axis a = r_p. Compare the orbits, mean anomalies, and the
velocity of Molniya_1 and CiRK_2. Discuss.

"""


def task_c():
    e_cirk = 0  # Eccentricity for circular orbit

    r_p_molniya = semi_major_axis*(1-eccentricity)
    print(f"Perigee: {(r_p_molniya-r_e)/1000} km from Earth surface")

    a_CiRCK_2 = r_p_molniya # m

    theta = np.linspace(0, 2*np.pi, 1000) # Angle vector

    # Function for calculating the radius and eccentric anomaly
    def r_and_E(semi_major_axis, eccentricity):
        E = 2*np.arctan(np.tan(theta/2)*np.sqrt((1-eccentricity)/(1+eccentricity))) # Eccentric anomaly
        radius = semi_major_axis*(1-eccentricity*np.cos(E))
        return radius

    # Calculating X, Y and Z position vectors for 3D plot
    X_m = r_and_E(semi_major_axis, eccentricity) * (np.cos(OMEGA)*np.cos(omega+theta)-np.sin(OMEGA)*np.cos(inclination)*np.sin(omega+theta))
    Y_m = r_and_E(semi_major_axis, eccentricity) * (np.sin(OMEGA)*np.cos(omega+theta)+np.cos(OMEGA)*np.cos(inclination)*np.sin(omega+theta))
    Z_m = r_and_E(semi_major_axis, eccentricity) * (np.sin(inclination)*np.sin(omega+theta))

    X_c = r_and_E(a_CiRCK_2, e_cirk) * (np.cos(OMEGA)*np.cos(omega+theta)-np.sin(OMEGA)*np.cos(inclination)*np.sin(omega+theta))
    Y_c = r_and_E(a_CiRCK_2, e_cirk) * (np.sin(OMEGA)*np.cos(omega+theta)+np.cos(OMEGA)*np.cos(inclination)*np.sin(omega+theta))
    Z_c = r_and_E(a_CiRCK_2, e_cirk) * (np.sin(inclination)*np.sin(omega+theta))

    # Calculating X and Y vectors for 2D plot
    X_m_2d = r_and_E(semi_major_axis, eccentricity) * np.cos(theta)
    Y_m_2d = r_and_E(semi_major_axis, eccentricity) * np.sin(theta)

    X_c_2d = r_and_E(a_CiRCK_2, e_cirk) * np.cos(theta)
    Y_c_2d = r_and_E(a_CiRCK_2, e_cirk) * np.sin(theta)

    # Converting to km
    X_m = X_m/1000
    Y_m = Y_m/1000
    Z_m = Z_m/1000

    X_c = X_c/1000
    Y_c = Y_c/1000
    Z_c = Z_c/1000
```

```
    X_m_2d = X_m_2d/1000
    Y_m_2d = Y_m_2d/1000

    X_c_2d = X_c_2d/1000
    Y_c_2d = Y_c_2d/1000

    earth_r = 6371 # km
    earth_theta = np.linspace(0, 2*np.pi, 1000) # Angle vector
    earth_X = earth_r * np.cos(earth_theta) # Radius
    earth_Y = earth_r * np.sin(earth_theta) # Radius
    earth_z = np.zeros(1000) # Radius


     # 2D plot of the orbit
    #plt.plot(X_m, Y_m, label='Molniya_1 orbit')
    plt.plot(X_m_2d, Y_m_2d, label='Molniya_1 orbit')
    plt.plot(X_c_2d, Y_c_2d, label='CiRK_2 orbit')
    plt.fill_between(earth_X, earth_Y, color='teal')
    plt.text(-3000, 0, 'Earth', fontsize=12, color='white')
    plt.xlabel('X [km]')
    plt.ylabel('Y [km]')
    plt.title('Molniya_1 orbit')
    plt.legend()
    plt.show()

    # 3D plot of the orbit
    fig = plt.figure()
    ax = plt.axes(projection='3d')
    ax.plot3D(X_m, Y_m, Z_m, label='Molniya_1 orbit')
    ax.plot3D(X_c, Y_c, Z_c, label='CiRK_2 orbit')
    ax.plot3D(earth_X, earth_Y, earth_z)
    ax.add_collection3d(plt.fill_between(earth_X, earth_Y, earth_z, color='teal'))
    ax.set_xlabel('X [km]')
    ax.set_ylabel('Y [km]')
    ax.set_zlabel('Z [km]')
    plt.title('Molniya_1 orbit')
    plt.legend()
    plt.show()

    # calculating the mean anomaly of molniya_1
    M_m = np.sqrt(mu/semi_major_axis**3)*time_span # Mean anomaly
    M_c = np.sqrt(mu/a_CiRCK_2**3)*time_span # Mean anomaly

    # calculating the velocity in perigee of molniya_1 and cirk_2
    v_m = np.sqrt(mu*(2/r_p_molniya - 1/semi_major_axis))
    v_c = np.sqrt(mu*(2/r_p_molniya - 1/a_CiRCK_2))

    # calculating the velocity in apogee of molniya_1 and cirk_2
    r_appogee_m = 2*semi_major_axis - r_p_molniya
    r_appogee_c = 2*a_CiRCK_2 - r_p_molniya

    v_a_m = np.sqrt(mu*(2/r_appogee_m - 1/semi_major_axis))
```

```
    v_a_c = np.sqrt(mu*(2/r_appogee_c - 1/a_CiRCK_2))

    print("------------Molniya_1------------")
    print(f"Velocity at Perigee for Molniya_1: {v_m:.6} m/s")
    print(f"Velocity at Apogee for Molniya_1: {v_a_m:.5} m/s")
    print("------------CiRK_2------------")
    print(f"Velocity at Perigee for CiRK_2: {v_c:.5} m/s")
    print(f"Velocity at Apogee for CiRK_2: {v_a_c:.5} m/s")

    # ploting the mean anomaly over time for the Molniya_1 orbit and the
    circular orbit
    plt.plot(time_span, M_m%(2*np.pi), label='Molniya_1')
    plt.plot(time_span, M_c%(2*np.pi), label='CiRK_2')
    plt.xlabel('Time [s]')
    plt.ylabel('Mean anomaly [rad]')
    plt.title('Mean anomaly over time')
    plt.legend()
    plt.show()




if __name__ == '__main__':
    #task_a()
    #task_b()
    task_c()
```

References

[1] Contributors to Wikimedia projects. Molniya orbit - Wikipedia, August 2023. [Online; accessed 15. Sep. 2023].