elelktron2

February 6, 2024

[1]:
```python
import math
a = 40708 # (km)
e = 0.8320
i = 61.00 # (degre)
rayon_terrestre = 6378 # (km)
mu = 398600 #constante de gravitation réduite GM (km^3/s^2)
w = 270
alpha = 360 / 86164
L_omega = 120
```

0.1 FONCTION NÉCESSAIRES

[2]:
```python
def calculate_time(v, periapsis=False):
    term1 = -math.sqrt(a**3 / mu) if periapsis else math.sqrt(a**3 / mu)
    term2 = math.asin(math.sqrt(1 - e**2) * math.sin(math.radians(v)) / (1 + e␣
↪* math.cos(math.radians(v))))
    term3 = e * math.sqrt(1 - e**2) * math.sin(math.radians(v)) / (1 + e * math.
↪cos(math.radians(v)))

    v_rad = math.radians(v)

    if -v_c <= v_rad <= v_c:
        t_p = term1 * (term2 - term3)

    elif v_c < v_rad < 2*math.pi - v_c:
        t_p = term1 * (math.pi - term2 - term3)

    elif v_rad > 2*math.pi - v_c:
        t_p = term1 * (2*math.pi + term2 - term3)

    elif -2*math.pi + v_c < v_rad < -v_c:
        t_p = term1 * (-math.pi - term2 - term3)

    elif v_rad < -2*math.pi + v_c:
        t_p = term1 * (-2*math.pi + term2 - term3)

    return t_p
```

```python
def calculate_la(v):
    v_rad = math.radians(v)
    w_rad = math.radians(w)
    i_rad = math.radians(i)

    la = math.degrees(math.asin(math.sin(w_rad + v_rad) * math.sin(i_rad)))
    return la

def calculate_L0(la, i, w, v):
    la_rad = math.radians(la)
    i_rad = math.radians(i)
    w_rad = math.radians(w)
    v_rad = math.radians(v)

    tan_value = math.tan(la_rad) / math.tan(i_rad)

    if -1 <= tan_value <= 1:
        L0 = math.degrees(math.asin(tan_value))
    else:
        if tan_value > 1:
            tan_value = 1
        elif tan_value < -1:
            tan_value = -1

        L0 = math.degrees(math.asin(tan_value))


    if -w_rad - math.radians(90) <= v_rad <= -w_rad + math.radians(90):
        pass
    elif -w_rad - math.radians(450) <= v_rad <= -w_rad - math.radians(270):
        L0 = -360 + L0
    elif -w_rad - math.radians(270) <= v_rad <= -w_rad - math.radians(90):
        L0 = -180 - L0
    elif -w_rad + math.radians(90) <= v_rad <= -w_rad + math.radians(270):
        L0 = 180 - L0
    elif -w_rad + math.radians(270) <= v_rad <= -w_rad + math.radians(450):
        L0 = 360 + L0
    elif -w_rad + math.radians(450) <= v_rad <= -w_rad + math.radians(630):
        L0 = 540 - L0
    else:
        L0 = None

    return L0
```

0.2 Question 1

Apoapsis and periapsis altitudes, z_A and z_P

$$r_A = a(1+e)$$
$$r_P = a(1-e)$$

[3]:
```
r_A = a*(1+e) #74577.056 km
r_P = a*(1-e) #6838.940
```

Radius, r_A et r_P

$$z_A = r_A - R_{terre}$$
$$z_P = r_P - R_{terre}$$

[4]:
```
z_A = r_A - rayon_terrestre #68199.05 km
z_P = r_P - rayon_terrestre #460.94 km
```

Orbital period T in seconds, and in *sexagesimal form (hours-min-sec)*

$$T = 2\pi\sqrt{\frac{a^3}{\mu}}$$

[5]:
```
T = 2*math.pi*(a**3/mu)**0.5 #81739.29 sec
T_sexagesimal = T/3600 #22 hours
```

Mean motion in rad/s and in rd/day

$$\omega_{mean} = \frac{T}{2\pi}$$

[6]:
```
mean_motion = T/(2*math.pi) #13009.21 rad/s
```

Velocities at apoapsis and periapsis, V_A and V_P

Nous avons,

$$r_A V_A = r_P V_P$$

et

$$W = \frac{V^2}{2} - \frac{\mu}{r} = -\frac{\mu}{2a}$$

$$\equiv \frac{V_P^2}{2} - \frac{\mu}{r_P} = -\frac{\mu}{2a}$$

$$\equiv \frac{V_P^2}{2} = \frac{\mu}{r_P} - \frac{\mu}{2a}$$

$$\equiv V_P = \sqrt{\frac{2\mu}{r_P} - \frac{\mu}{a}}$$

et

$$V_A = \frac{r_P V_P}{r_A}$$

[7]:
```
V_P = (((2*mu)/r_P) - (mu/a))**0.5 #10.33 km/s
V_A = (r_P*V_P)/r_A #0.95

print("Vitesse au periapsis:", round(V_P,2), "km/s")
print("Vitesse à l'apoapsis:", round(V_A,2), "km/s")
```

```
Vitesse au periapsis: 10.33 km/s
Vitesse à l'apoapsis: 0.95 km/s
```

Velocity Ratio

$$\frac{V_P}{V_A}$$

Ratio de vitesse égal à 1 : Lorsque le ratio de vitesse entre l'apoapsis et le periapsis est égal à 1, les vitesses à ces points sont identiques. Cela se produit dans le cas d'une orbite circulaire où l'apoapsis et le periapsis coïncident, donc la vitesse reste constante sur l'ensemble de l'orbite.

Ratio de vitesse supérieur à 1 : Si le ratio est supérieur à 1, cela signifie que la vitesse à l'apoapsis est plus grande que la vitesse au periapsis. Dans les orbites elliptiques, l'objet spatial se déplace plus rapidement à l'apoapsis, le point le plus éloigné de l'objet central (comme la Terre), et plus lentement au periapsis, le point le plus proche.

Ratio de vitesse inférieur à 1 : Si le ratio est inférieur à 1, cela indique que la vitesse au periapsis est plus grande que la vitesse à l'apoapsis. Dans ce cas, l'objet spatial se déplace plus rapidement au periapsis et plus lentement à l'apoapsis.

[8]:
```
ratio = V_P/V_A #10.90 > 1
```

0.3 Question 2

You will determine the ground track plot of this orbit. Coordinates of ascending node are following: Latitude 0° by definition obviously and Longitude 120° East-positive

[9]:
```
w = 270
```

Etape 1 : Determine the critical true anomaly

$$v_c = cos^{-1}(-e)$$

[10]:
```
v_c = math.acos(-e) #2.55 RAD !!!!!!
```

Etape 2 :

CAR NOUS AVONS i = 61° < 90° :

$$-v_c \leq v \leq v_c$$

ratoation par $\pi-$:

$$v_c \leq v \leq 2\pi - v_c$$

ratoation par $2\pi+$:

$$2\pi - v_c \leq v \leq 2\pi + v_c$$

ratoation par $3\pi-$:

$$2\pi + v_c \leq v \leq 4\pi - v_c$$

ratoation par $-2\pi+$:

$$-2\pi - v_c \leq v \leq -2\pi + v_c$$

ratoation par $-\pi-$:

$$-2\pi + v_c \leq v \leq -v_c$$

$$v = -\omega = -270° = -\frac{3}{2}\pi$$

SI i > 90° : $-\pi-$ => $-2\pi+$ => $-3\pi-$ => $3\pi-$ => $2\pi+$ => $\pi-$

[11]:
```
v = -1.5*math.pi
```

Etape 3 : Calcul de tp

$v < -2\pi + v_c$ **donc faut ajouter** $-2\pi+$ **dans l'equation de t-tp**

$$t_p = -\sqrt{\frac{a^3}{\mu}}[-2\pi + sin^{-1}(\frac{\sqrt{1-e^2}sin(v)}{1+ecos(v)}) - e\frac{\sqrt{1-e^2}sin(v)}{1+ecos(v)}]$$

[12]:
```
tp = calculate_time(v=-w,periapsis=True)
tp
```

[12]: 80093.3668829082

[13]:
```
t_p =  -math.sqrt(a**3 / mu) * (-2*math.pi + math.asin(math.sqrt(1 - e**2) *␣
↪math.sin(v) / (1 + e * math.cos(v))) - e * math.sqrt(1 - e**2) * math.sin(v)␣
↪/ (1 + e * math.cos(v)))
print("Temps periapsis =>", t_p)
```

```
Temps periapsis => 80093.3668829082
```

[14]:
```
v_angles = [ i for i in (range(-210,210,30))]
```

[15]:
```
t = []

for j in v_angles:
    t.append( round(calculate_time(v=j) + tp,2) )



La = []


for index, v in enumerate(v_angles):
    result_la = calculate_la(v)
    La.append(round(result_la,2))


L0 = [ ]

for index , v in enumerate ( v_angles ) :
    result_la = calculate_la ( v )
    result_L0 = calculate_L0 ( result_la , i , w , v )
    L0.append ( round(result_L0,2) )


Ls = []

for index,l in enumerate(L0) :
    ls = L_omega + l - alpha * t[index]
    Ls.append(round(ls,2))
```

[16]:
```
import pandas as pd

data = {
    'v_angles': v_angles,
    'times_at_periapsis': t,
    'la_angles': La,
    'l0_angles': L0,
    'Ls' : Ls,
}

# Create a DataFrame
df = pd.DataFrame(data)

# Transpose the DataFrame
df = df.T
```

```python
# Set the first row as the column headers
df.columns = df.iloc[0]
df = df[1:]

df
```

[16]:
```
v_angles               -210.0    -180.0    -150.0    -120.0     -90.0     -60.0   \
times_at_periapsis    9633.58  39223.72  68813.86  76429.36  78447.44  79262.40
la_angles               49.24     61.00     49.24     25.93      0.00    -25.93
l0_angles               40.02     90.00    139.98    164.36    180.00    195.64
Ls                     119.77     46.12    -27.53    -34.97    -27.76    -15.52

v_angles                -30.0       0.0      30.0      60.0      90.0   \
times_at_periapsis   79731.81  80093.37  80454.92  80924.33  81739.29
la_angles              -49.24    -61.00    -49.24    -25.93     -0.00
l0_angles              220.02    270.00    319.98    344.36    360.00
Ls                       6.89     55.36    103.83    126.25    138.49

v_angles                120.0     150.0      180.0
times_at_periapsis   83757.37  91372.87  120963.01
la_angles               25.93     49.24      61.00
l0_angles              375.64    400.02     450.00
Ls                     145.70    138.26      64.61
```