# Design of a new mission to Uranus

Student Hugo Lancery, Lucie Michelet
Professor Robert Vincent

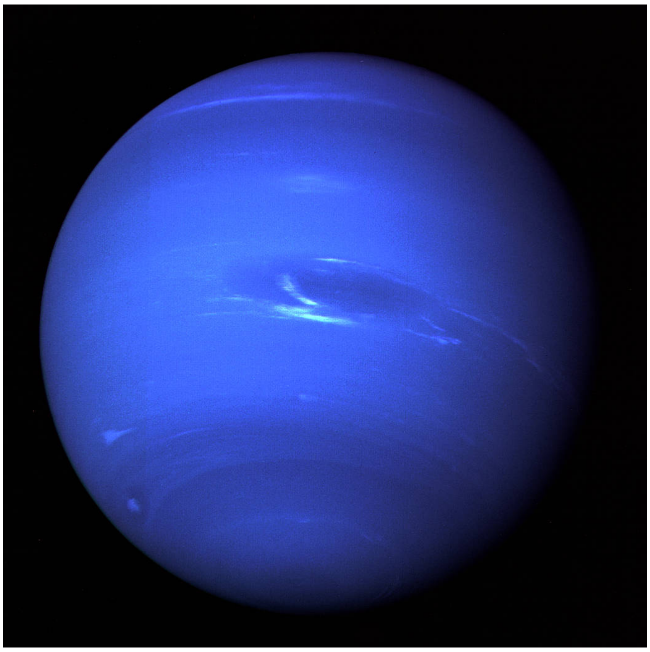
## Objectives

As generalities, the outer planets of the Solar System were targets for spacecrafts in the 60's and 70's, only. Since these first missions, the scientific interest has mainly been focused on Mars, Jupiter and Saturn. A White Paper has been published in fall 2021 by ESA to ask for scientific interests with Neptune and Uranus. This is the premise of a future mission, and space agencies ask your team to work in this framework to answer both following questions:

1. What would be the cost to for a new spacecraft to travel from Earth to Uranus ?

2. What would be the best period for such a mission ?

Mettre le screen de la nasa avec un petit recap



Avril 2023

# Contents

# 1 Analytics calculations

Direct application of the lectures to determine the interplanetary transfer from Earth to Uranus.

## 1.1 Formulas and python

Let's properly start by defining all the datas we have:

| Variable | Value | Defenition |
|---|---|---|
| $G$ | $6.674e^{-11}$ | Gravitational constant |
| $mT$ | $5.974e^{24}$ | Earth mass $(kg)$ |
| $mU$ | $8.681e^{25}$ | Uranus mass $(kg)$ |
| $\mu_E$ | 398701.5 | Earth g reduced |
| $\mu_U$ | 5793942.5 | Uranus g reduced |
| $\mu_S$ | $13.2e^{10}$ | Sun g reduced |
| $\mu_J$ | $12.65e^{7}$ | Jupyter g reduced |
| $s_J$ | $750e^{6}$ | Distance S-J $(km)$ |
| $s_U$ | $2.9e^{9}$ | Distance S-U $(km)$ |
| $s_T$ | $150e^{6}$ | Distance S-E $(km)$ |
| $\omega_p$ | 6678 | Periapsis $(km)$ |
| $V_0$ | 14.5 | Initial velocity $(m.s^{-1})$ |
| $\mu_E$ | $398.6e^{4}$ | G reduced $(km^3.s^{-2})$ |
| $r_J$ | 71400 | Radius Jupyter $(km)$ |
| $r_U$ | 25362 | Radius Uranus $(km)$ |
| $r_T$ | 6378 | Radius Earth $(km)$ |

Then we programmed some algorithms to calculate faster the regularly used formulas. We will explain bellow the details of the algorithm in our library.

The liberation velocity :

$$\boxed{V_l = \left(\frac{2\mu_E}{r}\right)} \tag{1}$$

```
def Vl(mue,d):
    return np.sqrt(2*mue/d)
```

The satelization velocity :

$$\boxed{V_{sat} = \left(\frac{\mu_E}{r}\right)} \tag{2}$$

```
def Vsat(mue,R):
    return np.sqrt(mue/R)
```

The rotation velocity :

$$\boxed{V_{rot} = \left(\frac{2\pi R}{T}\right)} \tag{3}$$

```
def Vrot(R,T):
    return 2*np.pi*R/T
```

The velocity of the spacecraft according to the Sun :

$$\boxed{V_s = \left(\sqrt{V_{\infty T}^2 + V_T^2 + 2V_{\infty T}V_T\cos\psi}\right)} \tag{4}$$

```
def Vs(V1,V2,angle):
    return np.sqrt(V1**2+V2**2+2*V1*V2*np.cos(angle))
```

The parameters : eccentricity (e), apoapsis (a), total energy (w) and time of revolution (T)

$$\boxed{a = \pm \frac{\mu}{2(\frac{V^2}{2 - \frac{\mu}{R}})}} \tag{5}$$

$$\boxed{e = \pm \frac{R}{a+1}} \tag{6}$$

$$\boxed{\omega = \pm \frac{\mu}{2a}} \tag{7}$$

$$\boxed{T = 2\pi \sqrt{\frac{a^3}{\mu}}} \tag{8}$$

The following code calculate each of them according to the parameter "h" hyperbola or "e" ellipse that the user enter.

```
def param(mue,V,R,mod):
    if mod == 'e' :
        a = -mue/(2*(V**2/2-mue/R))
        e = -R/a+1
        w = -mue/(2*a)
    if mod == 'h' :
        a = mue/(2*(V**2/2-mue/R))
        e = R/a+1
        w = mue/(2*a)
    t = 2*np.pi*np.sqrt(a**3/mue)
    return a,e,w,t
```

Eccentric anomaly and and time of an elliptic trajectory :

$$\boxed{E = \arccos(\frac{r}{a} - 1)\frac{1}{-e}} \tag{9}$$

$$\boxed{t = \sqrt{\frac{a^3}{\mu}}(E - e \sin E) + tp} \tag{10}$$

```
def E_e(R,a,e):
    return np.arccos((R/a-1)/(-e))

def t_tp_e(E,a,e,mue,tp=0):
    return (np.sqrt(a**3/mue))*(E - e*np.sin(E))+tp
```

Eccentric anomaly and and time of an hyperbolic trajectory :

$$\boxed{E = \cosh^{-1}(\frac{r}{a} + 1)\frac{1}{e}} \tag{11}$$

$$\boxed{t = \sqrt{\frac{a^3}{\mu}}(e \sinh E - E) + tp} \tag{12}$$

```
def E_h(R,a,e):
    return np.arccosh((R/a+1)/(e))

def t_tp_h(E,a,e,mue,tp=0):
    return (np.sqrt(a**3/mue))*(e*np.sinh(E)-E)+tp
```

Then we created a small code to compare two velocities :

```
def deltaV(V1,V2):
    if V2 > V1 :
        print("Thanks to Jupiter influence, we accelerated !")
        deltaV = V2-V1

    if V1 > V2 :
        print("Thanks to Jupiter influence, we decelerated !")
        deltaV = V1-V2

    if V1 == V2 :
        print("Jupiter influence didn't affect the velocity of the spacecraft.")
        deltaV = 0

    return deltaV
```

Now that the formulas are detailed, we can go through their application. We used all of the codes before to calculate the itinerary from Earth to Uranus, with a flyby at Jupiter.

## 1.2 Initial State : orbit around the Earth

We calculate the velocity of satelization and liberation of our satellite and its parameters for a given altitude of 200km.

```
vsat = (mueE/omgp)**0.5
vlib = Vl(mueE,omgp)

#parameters initial state
a = -mueE/(2*(vsat**2/2-mueE/omgp))
e = 0   # because LEO
t = 2*np.pi*np.sqrt(a**3/mueE)

print("Satelization velocity : ",vsat,"\nLiberation Velocity : ",vlib)
```

Satelization velocity : 7.785329191319328
Liberation Velocity : 11.010118129902954

## 1.3 First Tranfert Orbit - leaving Earth inlfuence

We can see that the velocity of our spacecraft is too low to leave the earth influence, so we add a first thruster injection. The objective is to have an hyperbolic trajectory to do the tranfert. So the eccentricity needs to be above 1.
The code bellow calculates the new velocity for an injection of 6.66 $km.s^{-1}$ and the parameter of the new trajectory. We chose to have a ray of 900000 km.

```
dv0 = 6.66
v0 = vsat + dv0
a0,e0,w0,T0 = param(mueE,v0,omgp,'h')

phi0 = np.arccos(1/e0)*180/np.pi

Ro = 900000
tp = 0

E0 = E_h(Ro,a0,e0)
E0deg = E0*180/np.pi

t0 = t_tp_h(E0,a0,e0,mueE,tp)
print(t0/60/60,'hours')
```

26.178537461802 hours

The table bellow shows the result of each parameter at stage 0 :

| Variable | Value | Defenition |
|---|---|---|
| $V_{sat_0}$ | 7.78 | Satellization velocity Earth ($km.s^{-1}$) |
| $V_{l_0}$ | 11 | Liberation velocity Earth ($km.s^{-1}$) |
| $V_0$ | 14.45 | Velocity at stage 0 ($km.s^{-1}$) |
| $a_0$ | 4559 | Semi grand axis ($km$) |
| $e_0$ | 2.45 | Eccentricity |
| $\omega_0$ | 43.72 | Total energy ($J$) |
| $T_0$ | 3063 | Time of rotation ($s$) |
| $phi0$ | 65.8 | Angle($deg$) |
| $E_0$ | 291 | Eccentric anomaly($deg$) |
| $t_0$ | 5308 | Time to leave earth influence($s$) |

We can see that the velocity of 14.5 km/s is high enough to leave earth influence and start its journey to the limit of Earth inlfuence in 5305 seconds.
Now we want to know what is the velocity needed to enter Jupiter influence !

So we calculate the velocity of the spacecraft at the limit of Earth influence.

```
#velocity at the limit of the earth influence sphere
VinfE = V(mueE,a0)
```

| Variable | Value | Defenition |
|---|---|---|
| $V_{\infty_E}$ | 9.35 | Spacecraft velocity at limit Earth influence ($km.s^{-1}$) |

Then we enter the Deep Space ! So we calculate the rotation velocity of the Earth :

```
ua = 150*10**6
TrotE = 365.25*86400

#velocity rotation earth
VrotE = Vrot(ua,TrotE)
```

| Variable | Value | Defenition |
|---|---|---|
| $V_{rot_E}$ | 29.86 | Earth rotation velocity ($km.s^{-1}$) |

By hypothesis, the heliocentric orbit is an ellipse with its periapsis at the level of Earth orbit. The slope of V0 the velocity related to Sun at the exit point, is measured from the Earth motion vector at 19,53°.

```
angle_V = 19.53*np.pi/180

#injection velocity
Vso = Vs(VinfE,VrotE,angle_V)
Vl1 = Vl(mueS,ua)
# Vl1 > Vso so we have an ellipse

print("Velocity needed : ",Vl1)
print("Velocity we have : ",Vso)
```

Velocity needed : 41.95235392680606
Velocity we have : 38.80463211137637

Then we calculate the corresponding parameters and the time of transfert to Jupiter influence sphere :

```
#The parameter of the trajectory once the injection is done
a1,e1,w1,T1 = param(mueS,Vso,ua,'e')

E1 = E_e(sJ,a1,e1)
t1 = t_tp_e(E1,a1,e1,mueS)
print("Time of transfer : ",t1/60/60/24/365,'year')
```

Time of transfer : 1.745828651854084 year

| Variable | Value | Defenition |
|---|---|---|
| $V_{l_1}$ | 41.95 | Liberation velocity at stage 1 ($km.s^{-1}$) |
| $V_{s_0}$ | 38.8 | Spacecraft injection velocity ($km.s^{-1}$) |
| $a_1$ | 519275084 | Semi grand axis ($km$) |
| $e_1$ | 0.71 | Eccentricity |
| $\omega_1$ | -127 | Total energy ($J$) |
| $T_1$ | 204639341 | Time of rotation ($s$) |
| $E_1$ | 2.24 | Eccentric anomaly($deg$) |
| $t_1$ | 55056452 | Time to enter Jupiter influence($s$) |

We left the Earth on June 07, 2034. So it gives a date of rendez-vous with Jupiter on March 08, 2036 as the NASA website planned it.

## 1.4 Arrival in the Jupiter neighborhood

To prepare the arrival in the Jovian sphere we need to know the rotation velocity of the planet :

```
# Velocity jupiter related to the sun
Vj = V(mueS,sJ)
print("Velocity of Jupiter : ",Vj)
```

Velocity of Jupiter : 13.2664991614216 km/s

Then we calculate the velocity of the spacecraft, related to Sun, at the level of Jupiter orbit and the angle of crossing orbit directions :

```
Vs1 = np.sqrt(2*(-mueS/(2*a1)+mueS/sJ))
phi1 = np.arccos(Vso*ua/(Vs1*sJ))
phi1deg = phi1*180/np.pi
print(Vs1)
```

Vs1 : 9.889361622433585 km/s
Phi1 : 38.3 °

Then we calculate the velocity of the spacecraft, related to Jupiter, at the entry point and the angle :

```
Vinf1 = np.sqrt(Vs1**2+Vj**2-2*Vs1*Vj*np.cos(phi1))
alpha1 = np.arcsin(np.sin(phi1)*Vs1/Vinf1)
alpha1deg = np.rad2deg(alpha1)

print(Vinf1)
```

Vinf1 = 8.238860695103597 km/s
alpha1 = 48 °

Now We determine the following parameters of the flyby trajectory with 18.03 radii altitude (which is 18x the ray of Jupiter) according to the NASA data :

```
h1 = 18.03*rJ
Rj = rJ+h1

aj = mueJ/(Vinf1**2)
ej = 1+Rj/aj
phij = np.arccos(1/ej)
phijdeg = phij*180/np.pi

alphajdeg = 180-2*phijdeg-alpha1deg
alphaj = alphajdeg*np.pi/180
```

We resume the parameters in the table bellow :

| Variable | Value | Defenition |
|---|---|---|
| $a_j$ | 1863615 | Semi grand axis ($km$) |
| $e_j$ | 1.73 | Eccentricity |
| $phi_j$ | 54.64 | Angle ($deg$) |
| $alpha_j$ | 22.59 | Angle($deg$) |

Now we want to determine the exit conditions of the Jovian sphere :

```
#velocity of the spacecraft, related to Sun, at the Jovian exit point and its angle
Vinf2 = Vinf1

Vs2 = np.sqrt(Vinf2**2+Vj**2+2*Vinf2*Vj*np.cos(alphaj))
phi2 = np.arccos(1/(1+Rj/(mueJ/Vs2**2)))

dV = deltaV(Vs1,Vs2)

print("\nVelocity at the Jovian exit point : ",Vs2,"\nPhi angle : ",phi2*180/np.pi)
```

Thanks to Jupiter influence, we accelerated !

Velocity at the Jovian exit point : 21.111492415859036
Phi angle : 80.0496791082567

The NASA planned a velocity of 15.74 km/s at this stage but we can't theoretically afford a lower velocity with an altitude of 18.03 radii. The angle is also very high, we where expecting 32.5°. To have the same velocity and angle than the NASA, we would have to lower the altitude to at least 1.5 radii.

## 1.5 Second Transfert Orbit : leaving Jupiter influence

By hypothesis, the heliocentric orbit at the exit point of Jupiter influence sphere are:
r = sJ
v = Vs2
phi = phi2

We determine the parameters of the new orbit :

```
Vl2 = Vl(mueS,sJ)
print("Velocity needed to leave Jupiter influence :",Vl2)
```

Velocity needed to leave Jupiter influence : 18.76166303929372

The liberation velocity is lower than the velocity of the spacecraft so we already have an hyperbola. We still need to add velocity to correct the trajectory and reach Uranus influence.

```
dv2 = -0.94
Vs3 = Vs2 + dv2

#Vs3>Vl2
a3,e3,w3,T3 = param(mueS,Vs3,sJ,'h')

print(a3)
```

2404848774 km

We calculate the time of rendez vous with Uranus

```
E3 = E_h(sU,a3,e3)
t3 = t_tp_h(E3,a3,e3,mueS)

print(t3/60/60/24/365,"years")
```

6.831969094364262 years

## 1.6 Total Time

We can conclude on the total time needed to go from Earth to Uranus :

```
    print((t0+t1+t3)/60/60/24/365,"years")
```

8.580786163736816 years

## 1.7 Arrival on Uranus

Now we want to have an orbit around an orbit around Uranus. We define the the orbit altitude at 2000 km :

```
h3 = 2000
Ru = rU+h3
```

Then we calculate the velocity of Uranus :

```
Vu = V(mueS,sU)
print(Vu)
```

6.7466466766320545 km/s

Then we need to know the velocity required to satellize our spacecraft around uranus at this distance :

```
VsatU = np.sqrt(mueU/Ru)
print("Vsat Uranus :",VsatU)
```

Vsat Uranus : 14.551681145658058

We calculate the liberation velocity of Uranus at this altitude, which is the velocity to not exceed :

```
VlU = Vl(mueU,Ru)
print(VlU)
```

20.579184831518486 km/s

We calculate the velocity of our spacecraft, related to Sun, at the level of Uranus orbit and the angle of crossing orbit directions :

```
Vs4 = V(mueS,a3)
phi3 = np.arccos(Vs3*sJ/(Vs4*sU))
phi3deg = phi3*180/np.pi

print(Vs4)
```

7.408718261822253 km/s

Now we calculate the velocity of the spacecraft, related to Uranus, at the entry point and the angle

```
Vinf4 = Vs(Vs4,Vu,phi3)
print(Vinf4)
```

13.068959099333043

We see that this velocity is too low to allow an elliptic orbit around Uranus, so we need to do an injection velocity. The NASA provides an injection of 2.11 km/s :

```
dv3 = 2.11
Vs5 = Vinf4+dv3
aU,eU,wU,Tu = param(mueU,Vs5,Ru,'e')

print("Vs5 :",Vs5,"\neU :",eU)
```

Vs5 : 15.178959099333042
eU : 0.08807201768712558

With this new velocity we are almost in a cylindric orbit which is perfect ! The velocity of the stalletite is between the satellization velocity and the liberation velocity. We are in orbit around Uranus.

## 1.8 Calculation of the cost

To calculate the cost, we need to know the total injection of velocity done during our journey :

```
dvTot = dv0+dv2+dv3
print(dvTot)
```

7.83 km/s

We have a final deltaV of 7.83 km/s. The NASA calculated 8.81 km/s, so we are 1km/s bellow the expected value.
To have the cost in properlant, we take average parameters for the mass and isp of a satellite using liquid propelant:

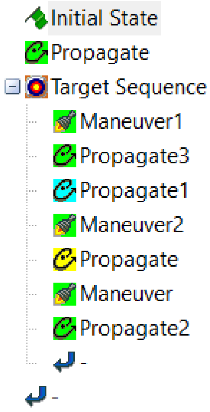
```
mi = 100
ISP = 200

mf = mi/np.exp(abs(dvTot)*1000/(9.81*ISP))
print(mi-mf,"kg")
```

98.15155545690361 kg

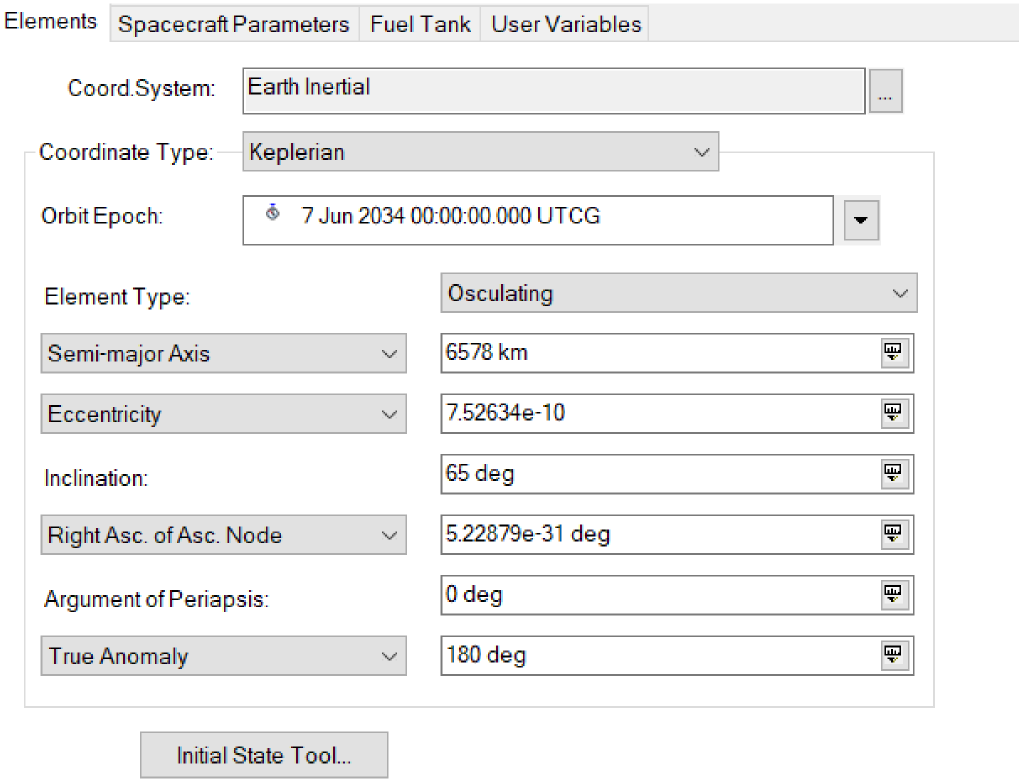
We finally have 98 percent of the mass that were used during this itinerary.
Now we will simulate this mission in STK and compare it to our theorical values. We will determine the inconsistencies and the differencies.

# 2 Simulation in STK

We now tried to implement everything in STK with what we calculated based on the nasa datas. The initial state:




We then apply those maneuvers, to:

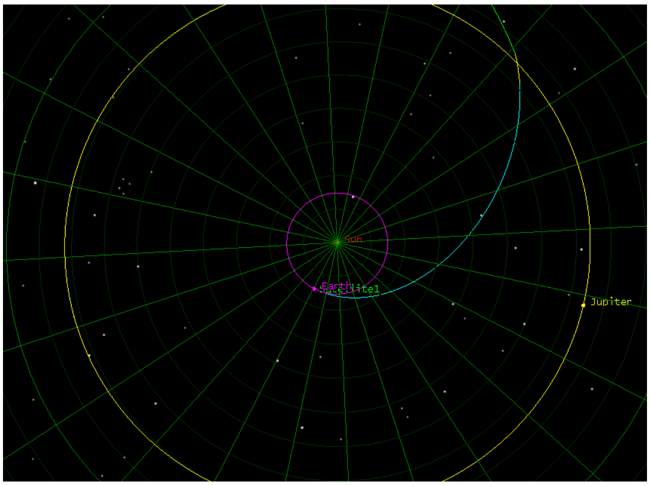



1.Orbit around the Earth
2.Maneuver on the VNC(Earth) axis X with a $\delta V = 6.655 km.s^{-1}$
3.Orbit in hyperbola in Earth influence
4.Orbit in ellipse in Heliocentric mode to jupyter
5.Maneuver on the VNC(Earth) axis X with a $\delta V = -0.8 km.s^{-1}$
6.Orbit in hyperbola in Jupyter influence
7.Orbit in ellipse in Heliocentric mode to uranus
8.Maneuver on the VNC(Earth) axis X with a $\delta V = 6.6 km.s^{-1}$
9.Rendez-vous with uranus

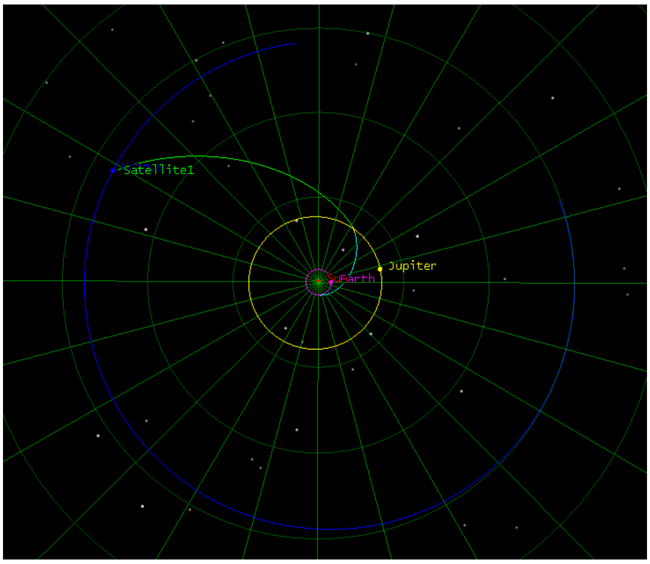
Our simulation finally gives us a total travel time from the 7 of june 2034 to the 25 of september 2046.
So approximately 12 years and 3 months.
With a global total injection of $\delta V = 13.255$ with a satellite using liquid propelant with initial 100kg we get 99.719% properlant used wich is totally realistic and optimized, but the travel time is still a bit too long.

## 2.1 Simulation representation

### 2.1.1 Orbit (1).(2).(3).(4).(5).(6)




### 2.1.2 All the orbits



# 3 Conclusion

As expected this project was very rich and allowed us to look more deeply what we saw and learned in class. We faced many problems on the coding and calculating first part of the project, since many equation resolution has to be made and on the STK part where a good knowledge of the software is undeniable essential to achieve the objectif. The most important things we learned is the importance of well defining the needs of a project to achieve the goal, and must of all the importance of the precision. In this mission the slightest 1s or 1° change can ruined and changes the trajectory due to a "butterfly" effect. To answer our first question, we managed to get $98.15kg$ and $99.719kg$ properlant used for theorical and STK, so we can expect that 100kg is not far from the reality of what this type of mission should cost. To answer our second question, the best period for this mission is very dependant of the departure day of the mission. We got with the theorical part 8.5807 years and with the STK 12.254 years.

IPSA
ÉCOLE D'INGÉNIEURS
DE L'AIR, DE L'ESPACE ET DE LA MOBILITÉ DURABLE