

Curso Dinamica Orbital 2025, TGallardo, FCien

```
In [57]: # determinacion de elementos orbitales a partir de r y v
from numpy import *
# vector posicion en uas y vector velocidad en uas/dia
pos = array([4.53377,1.93230,-0.10954])
vel = array([-1.12006,2.67368,0.0140043])/365.25
# instante en dias julianos
t=0.0
# masa del objeto en masas solares
m = 0.0009548
# masa estrella
Ms = 1
k = 0.01720209895
mu = k*k*(Ms + m)
```

```
In [58]: # signo de vel radial
print(dot(pos,vel))
```

0.00023743574750992622

```
In [59]: # modulos r y v
r = sqrt(dot(pos,pos))
print("r = ",r, "ua")
v2 = dot(vel,vel)
v = sqrt(v2)
```

r = 4.9295895077075125 ua

```
In [60]: # vector h
vech = cross(pos,vel)
hx = vech[0]
hy = vech[1]
hz = vech[2]
# modulo de h
h = sqrt(dot(vech,vech))
print("h = ",h)

# el vector excentricidad colineal con P
vex = cross(vel,vech)/mu - pos/r
ex = vex[0]
ey = vex[1]
ez = vex[2]
e = sqrt(dot(vex,vex))
a = 1/(2/r-v2/mu)
print("a = ",a, " ua")
print("e = ",e)
```

h = 0.039123468124387
a = 5.1799958149350385 ua
e = 0.048719591704451476

```
In [61]: rv = dot(pos,vel)
inc = arccos(hz/h)
print("inclinacion = ",inc*180/pi," grados")
vecz = array([0,0,1])
n1 = cross(vecz,vech)
# versor linea nodos
n = n1/sqrt(dot(n1,n1))
nx = n[0]
ny = n[1]
nod2 = arctan2(ny,nx)
print("longitud del nodo = ",nod2*180/pi," grados")
```

inclinacion = 1.3046833201070165 grados
longitud del nodo = 100.48313071118518 grados

```
In [62]: cosenow = dot(n,vex)/e
senow = ez/e/sin(inc)
w = arctan2(senow,cosenow)
print("argumento de perihelio = ",w*180/pi," grados")
```

argumento de perihelio = -84.90588389067435 grados

```
In [63]: # parametro de la conica
p = a*(1-e*e)
p
```

Out[63]: 5.1677005840386085

```
In [64]: cosenoverd=(p/r-1)/e
senoverd=dot(vex, cross(pos, vech))/(r*e*h)
averd=arctan2(senoverd,cosenoverd)
print("anom verdadera f =",averd*180/pi, "grados")
```

anom verdadera f = 7.503370931799901 grados

```
In [65]: cosaex=(1-r/a)/e

if a > 0:
    senaex=rv/(e*sqrt(a*mu))
    AE=arctan2(senaex, cosaex)
    AM=AE-e*sin(AE)
    T = t - AM*sqrt(a**3 / mu)
else:
    Z=cosaex
    F=log(Z+sqrt(Z*Z-1))*rv/abs(rv)
    T = t - (e*sinh(F)-F)*sqrt(-a**3 / mu)
print("instante periastro T =",T, "dias julianos")
```

instante periastro T = -81.29945354060277 dias julianos

In []:

In []: