Séparation isotopique par résonance cyclotron

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
```

1 Champ magnétique seul

[2]:
```
B0=0.5
uma=1.66e-27
m238,m235=238*uma,235*uma
e=1.6e-19
wc238=e*B0/m238
wc235=e*B0/m235
v0=np.sqrt(2*10*e/m238)
v1=np.sqrt(2*5*e/m238)
```

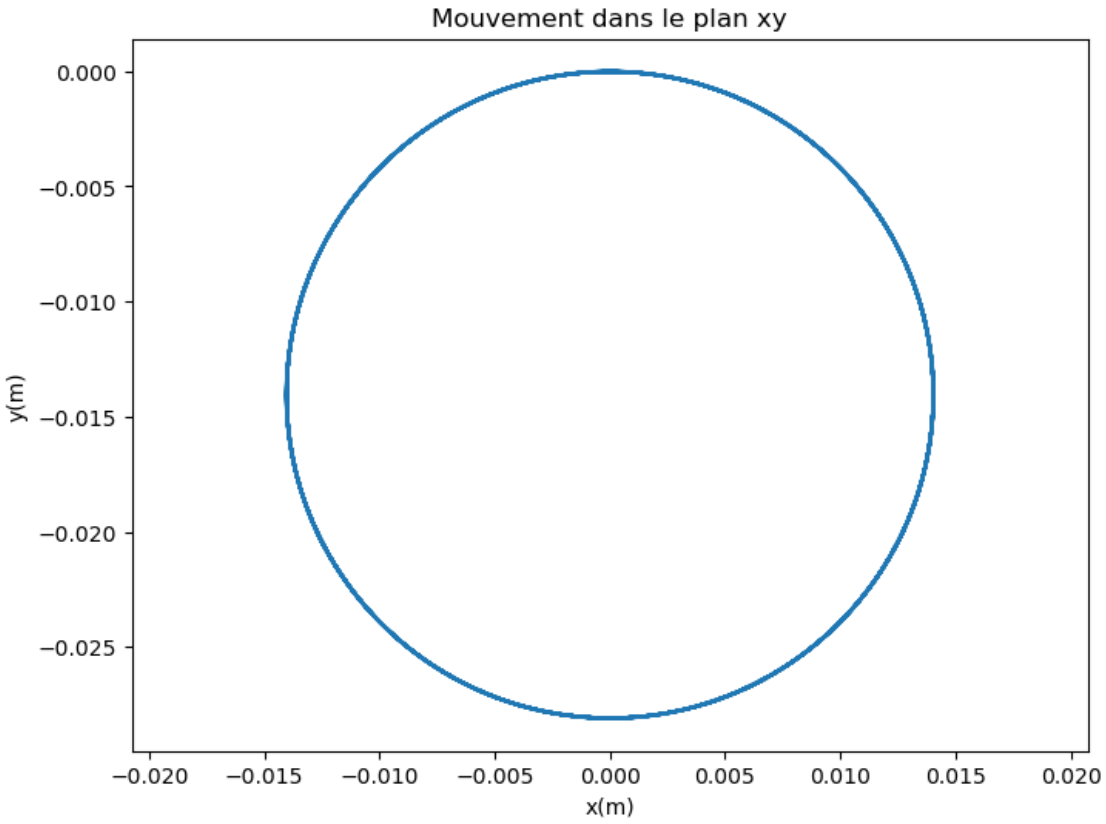
On résout les équations:

$$\begin{cases} \ddot{x} = \omega_c \dot{y} \\ \ddot{y} = -\omega_c \dot{x} + \dfrac{e.E(t)}{m} \\ \ddot{z} = 0 \end{cases}$$

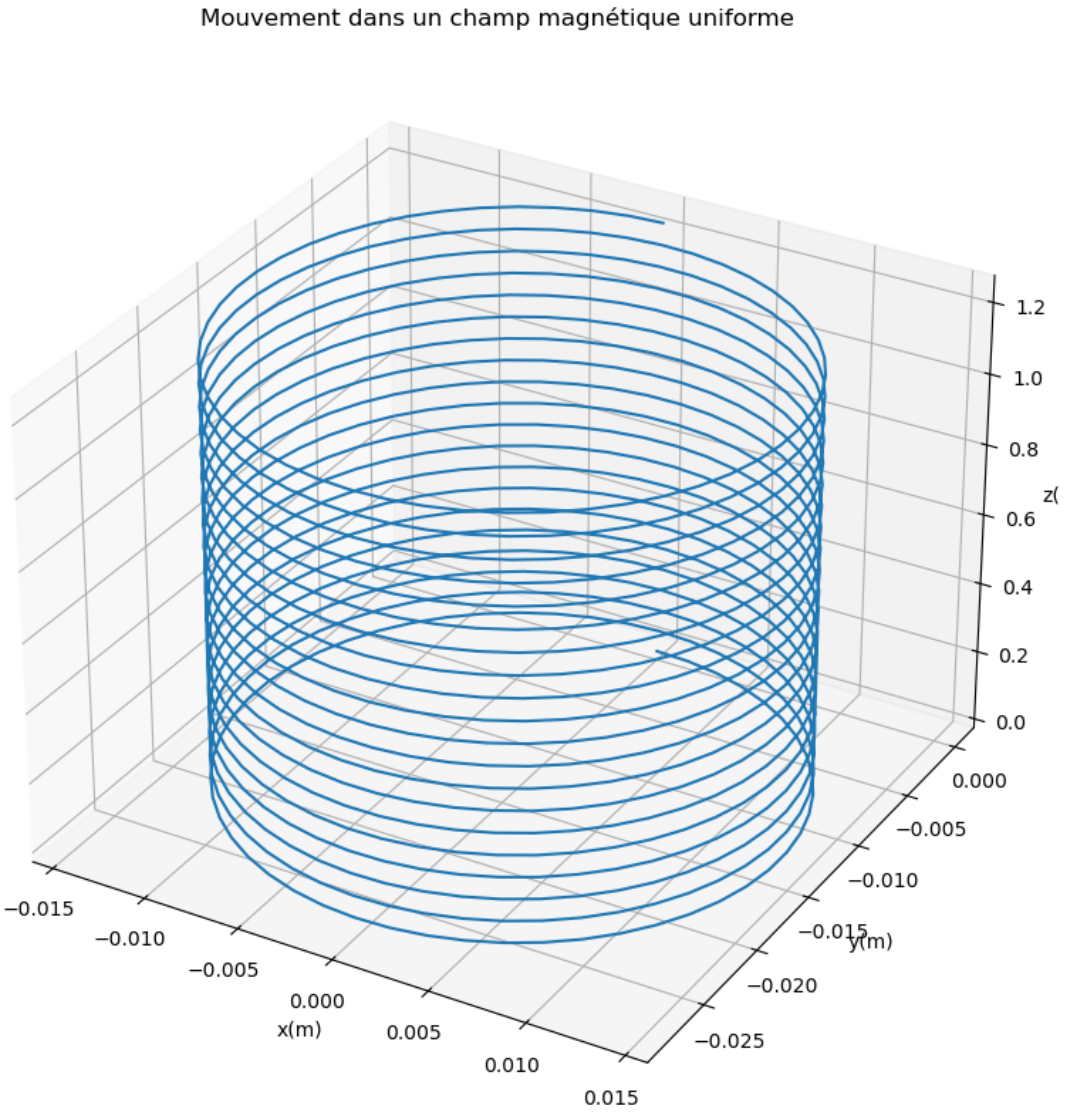
avec $\omega_c = \frac{eB_0}{m}$

[13]:
```
def derivee(inconnues,t):
    x=inconnues[0]
    dxdt=inconnues[1]
    y=inconnues[2]
    dydt=inconnues[3]
    z=inconnues[4]
    dzdt=inconnues[5]
    ddxdtt=wc238*dydt
    ddydtt=-wc238*dxdt
    ddzdtt=0
    return [dxdt,ddxdtt,dydt,ddydtt,dzdt,ddzdtt]
Tc=2*np.pi/wc238
tab_t=np.linspace(0,20*Tc,1000)
ci=[0,v0,0,0,0,v1]
sol=odeint(derivee,ci,tab_t)
```

```
tab_x=sol[:,0]
tab_y=sol[:,2]
tab_z=sol[:,4]
plt.figure(figsize=(8,6))
plt.axis('equal')
plt.plot(tab_x,tab_y)
plt.xlabel("x(m)")
plt.ylabel("y(m)")
plt.title("Mouvement dans le plan xy")
plt.show()
```



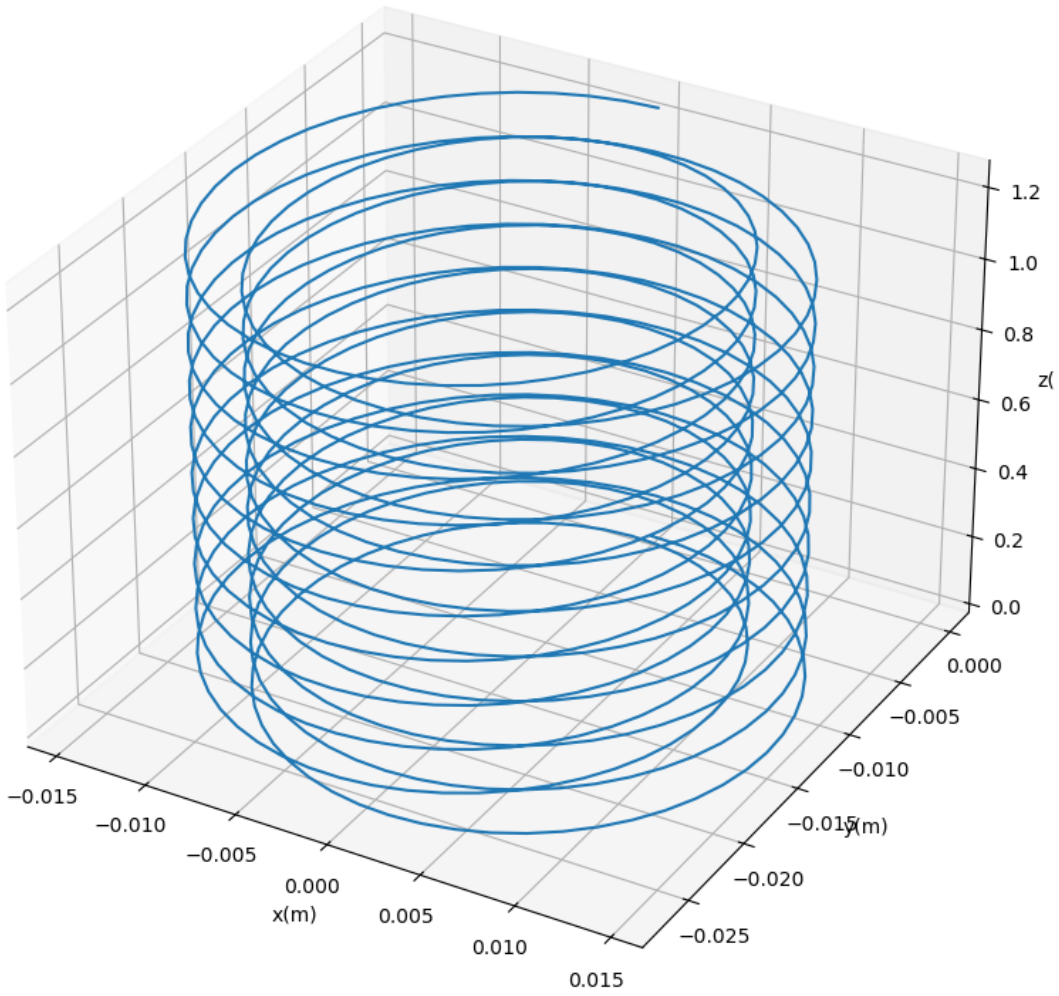
[19]:
```
from mpl_toolkits.mplot3d import Axes3D
fig=plt.figure(figsize=(10,10))
ax = fig.add_subplot(111, projection='3d')
ax.plot(tab_x,tab_y,tab_z)
ax.set_xlabel("x(m)")
ax.set_ylabel("y(m)")
ax.set_zlabel("z(m)")
plt.title("Mouvement dans un champ magnétique uniforme")
plt.show()
```



2 Résonance cyclotron

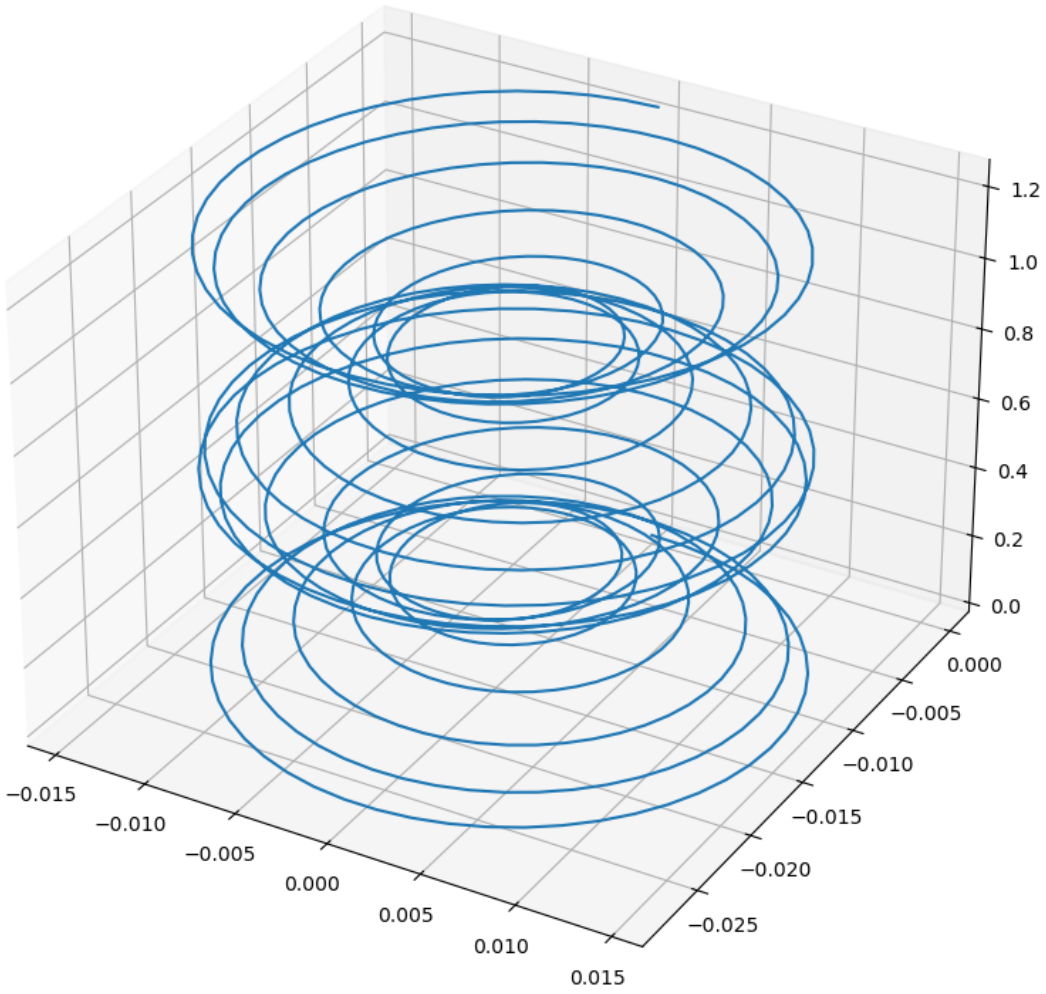
[34]:
```
m=m238
wc=wc238
E0=80
w=0.5*wc
def derivee(inconnues,t):
    x=inconnues[0]
    dxdt=inconnues[1]
    y=inconnues[2]
    dydt=inconnues[3]
```

```
    z=inconnues[4]
    dzdt=inconnues[5]
    ddxdtt=wc*dydt
    ddydtt=-wc*dxdt+e*E0*np.cos(w*t)/m
    ddzdtt=0
    return [dxdt,ddxdtt,dydt,ddydtt,dzdt,ddzdtt]
Tc=2*np.pi/wc
tab_t=np.linspace(0,20*Tc,1000)
ci=[0,v0,0,0,0,v1]
sol=odeint(derivee,ci,tab_t)
tab_x=sol[:,0]
tab_y=sol[:,2]
tab_z=sol[:,4]
from mpl_toolkits.mplot3d import Axes3D
fig=plt.figure(figsize=(10,10))
ax = fig.add_subplot(111, projection='3d')
ax.plot(tab_x,tab_y,tab_z)
ax.set_xlabel("x(m)")
ax.set_ylabel("y(m)")
ax.set_zlabel("z(m)")
plt.title(r"Mouvement dans un champ magnétique + électrique oscillant à␣
 ↪$\omega_c/2$")
plt.show()
```



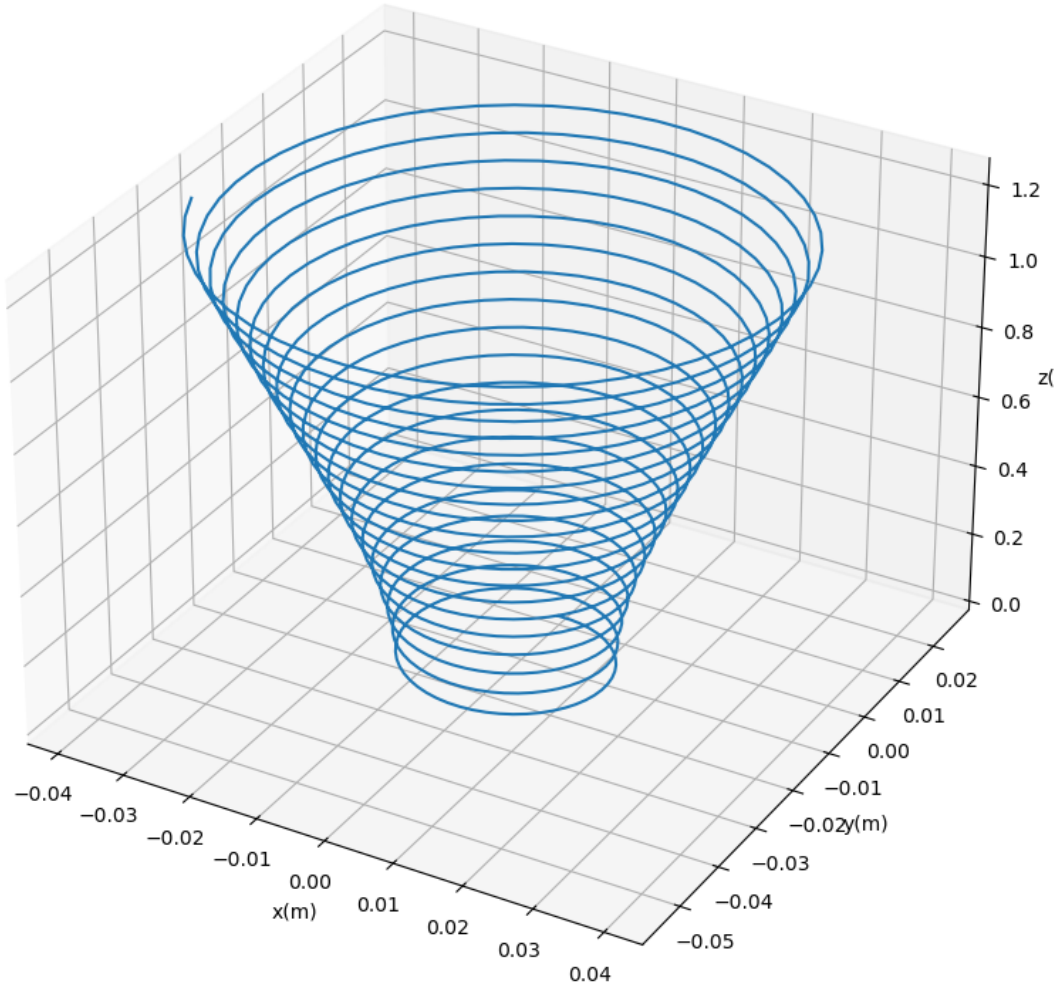
[31]:
```
m=m238
wc=wc238
E0=80
w=0.9*wc
Tc=2*np.pi/wc
tab_t=np.linspace(0,20*Tc,1000)
ci=[0,v0,0,0,0,v1]
sol=odeint(derivee,ci,tab_t)
tab_x=sol[:,0]
tab_y=sol[:,2]
tab_z=sol[:,4]
from mpl_toolkits.mplot3d import Axes3D
```

```
fig=plt.figure(figsize=(10,10))
ax = fig.add_subplot(111, projection='3d')
ax.plot(tab_x,tab_y,tab_z)
plt.show()
```

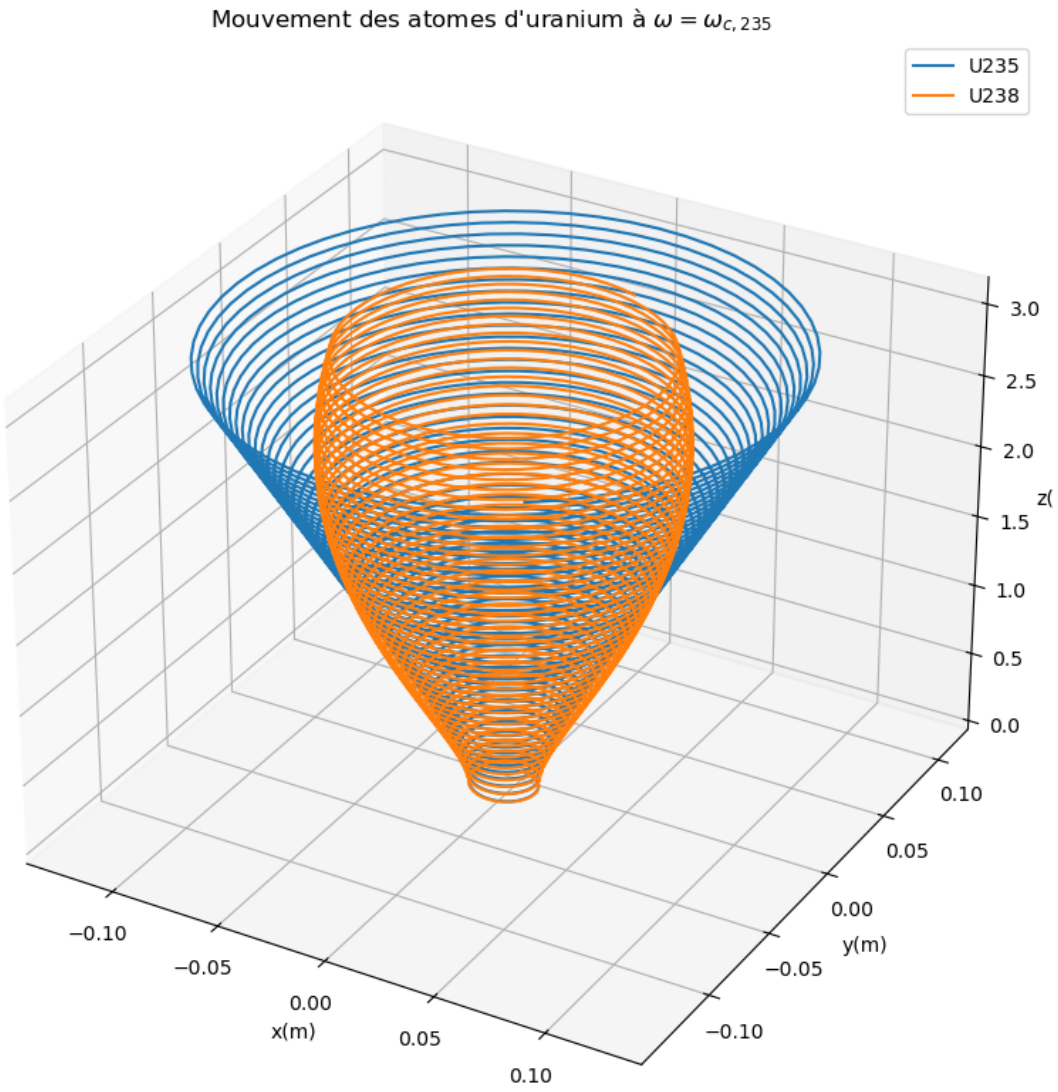


[32]:
```
m=m238
wc=wc238
E0=80
w=0.99*wc
Tc=2*np.pi/wc
tab_t=np.linspace(0,20*Tc,1000)
ci=[0,v0,0,0,0,v1]
sol=odeint(derivee,ci,tab_t)
```

```
tab_x=sol[:,0]
tab_y=sol[:,2]
tab_z=sol[:,4]
from mpl_toolkits.mplot3d import Axes3D
fig=plt.figure(figsize=(10,10))
ax = fig.add_subplot(111, projection='3d')
ax.plot(tab_x,tab_y,tab_z)
ax.set_xlabel("x(m)")
ax.set_ylabel("y(m)")
ax.set_zlabel("z(m)")
plt.title(r"Mouvement dans un champ magnétique + électrique oscillant à $0,99␣
 ↪\omega_c$")
plt.show()
```

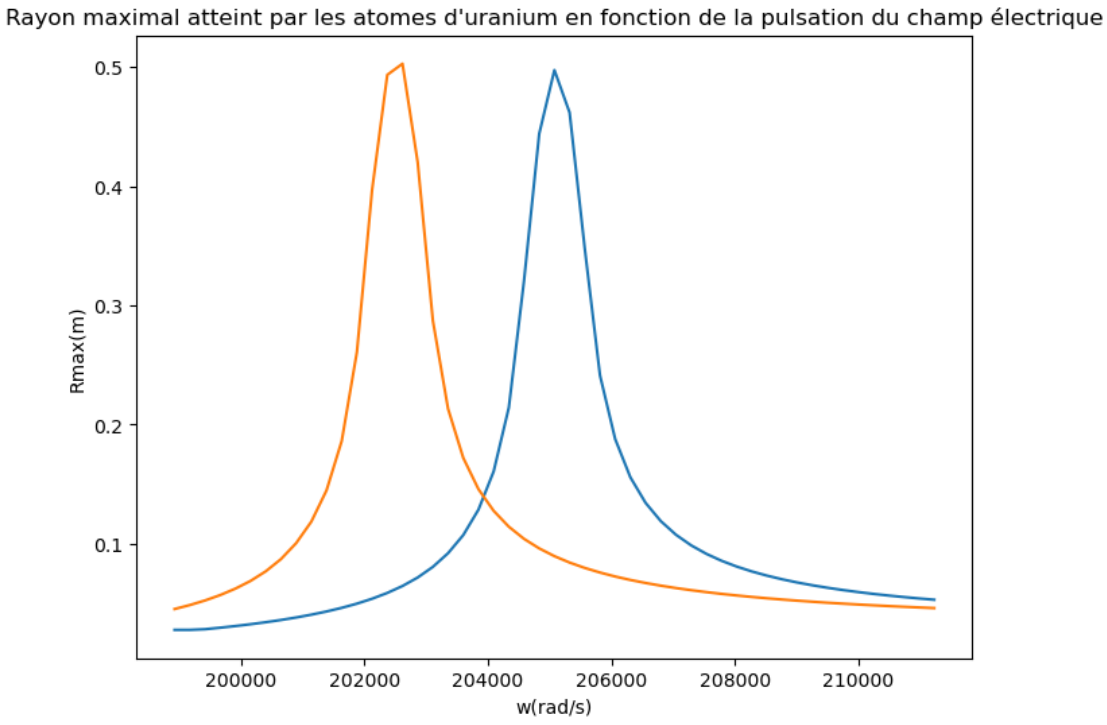


[25]:
```
m=m238
wc=wc238
E0=80
w=wc235
Tc=2*np.pi/wc
tab_t=np.linspace(0,50*Tc,3000)
ci=[0,v0,0,0,0,v1]
sol=odeint(derivee,ci,tab_t)
tab_x238=sol[:,0]
tab_y238=sol[:,2]
tab_z238=sol[:,4]
m=m235
wc=wc235
sol=odeint(derivee,ci,tab_t)
tab_x235=sol[:,0]
tab_y235=sol[:,2]
tab_z235=sol[:,4]
from mpl_toolkits.mplot3d import Axes3D
fig=plt.figure(figsize=(10,10))
ax = fig.add_subplot(111, projection='3d')
ax.plot(tab_x235,tab_y235,tab_z235,label="U235")
ax.plot(tab_x238,tab_y238,tab_z238,label="U238")
plt.legend()
ax.set_xlabel("x(m)")
ax.set_ylabel("y(m)")
ax.set_zlabel("z(m)")
plt.title(r"Mouvement des atomes d'uranium à $\omega=\omega_{c,235}$")
plt.show()
```



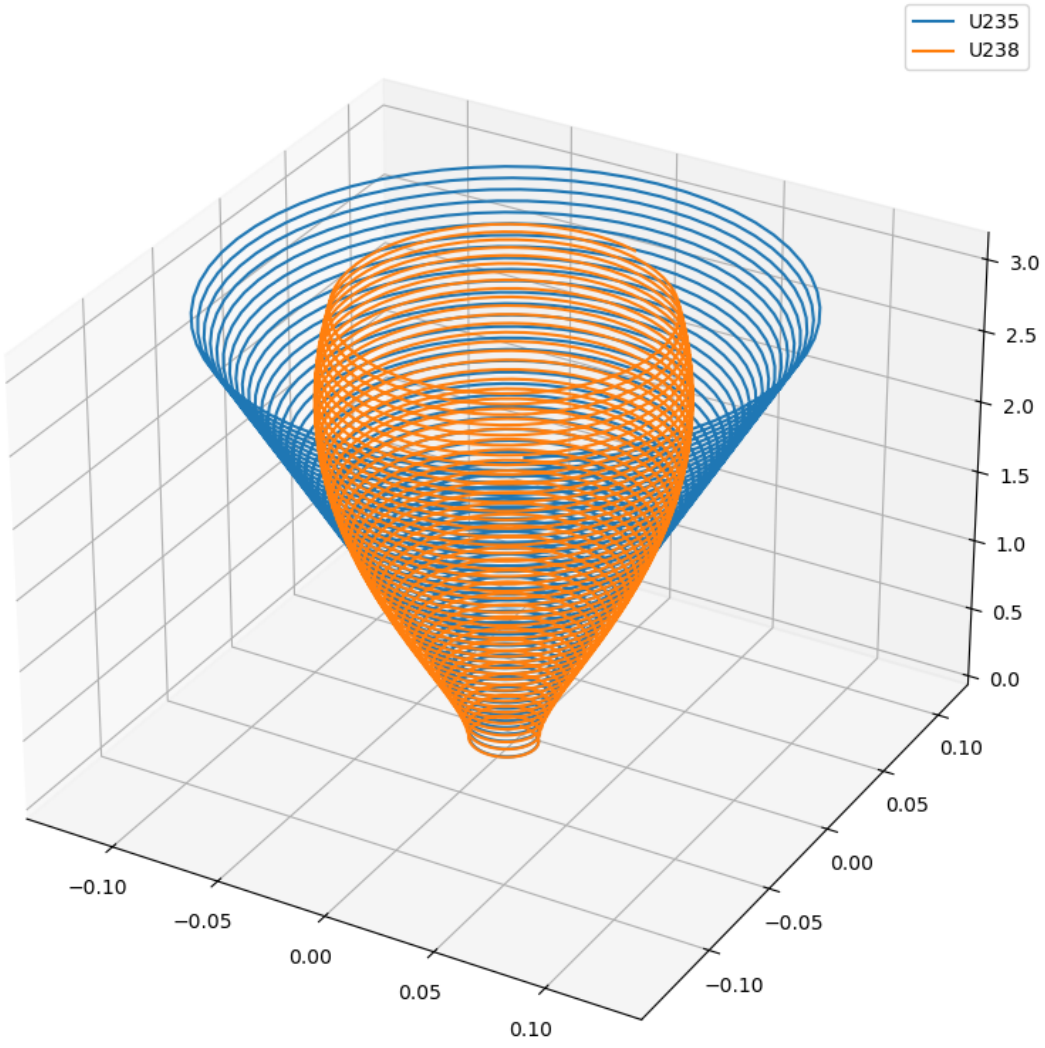
[36]:
```
def Rmax(w,wc):
    def derivee(inconnues,t):
        x=inconnues[0]
        dxdt=inconnues[1]
        y=inconnues[2]
        dydt=inconnues[3]
        z=inconnues[4]
        dzdt=inconnues[5]
        ddxdtt=wc*dydt
        ddydtt=-wc*dxdt+e*E0*np.cos(w*t)/m
        ddzdtt=0
        return [dxdt,ddxdtt,dydt,ddydtt,dzdt,ddzdtt]
```

```
    Tc=2*np.pi/wc
    tab_t=np.linspace(0,200*Tc,3000)
    ci=[0,v0,0,0,0,v1]
    sol=odeint(derivee,ci,tab_t)
    tab_x=sol[:,0]
    tab_y=sol[:,2]
    tab_r=np.sqrt(tab_x**2+tab_y**2)
    return max(tab_r)
tab_w=np.linspace(wc235*0.97,wc235*1.03,51)
tab_Rmax235=np.zeros_like(tab_w)
tab_Rmax238=np.zeros_like(tab_w)
for i in range(len(tab_w)):
    w=tab_w[i]
    tab_Rmax235[i]=Rmax(w,wc235)
    tab_Rmax238[i]=Rmax(w,wc238)
plt.figure(figsize=(8,6))
plt.plot(tab_w,tab_Rmax235,label="U235")
plt.plot(tab_w,tab_Rmax238,label="U238")
plt.xlabel("w(rad/s)")
plt.ylabel("Rmax(m)")
plt.title("Rayon maximal atteint par les atomes d'uranium en fonction de la␣
 ↪pulsation du champ électrique")
plt.show()
```



On peut montrer qu'une charge accélérée rayonne de la lumière et perd de l'énergie. C'est équivalent à ajouter une force de frottements approximativement égale à $-\frac{e^2 v}{6\pi\epsilon_0 c^3 r}$

[11]:
```
m=m238
wc=wc238
E0=80
w=wc235
e0,c=8e-12,3e8
K=e**2/(6*np.pi*e0*c**3)
def derivee(inconnues,t):
    x=inconnues[0]
    dxdt=inconnues[1]
    y=inconnues[2]
    dydt=inconnues[3]
    z=inconnues[4]
    dzdt=inconnues[5]
    r=np.sqrt(dxdt**2+dydt**2)/wc
    ddxdtt=wc*dydt-K*dxdt/r
    ddydtt=-wc*dxdt+e*E0*np.cos(w*t)/m-K*dydt/r
    ddzdtt=0
    return [dxdt,ddxdtt,dydt,ddydtt,dzdt,ddzdtt]
Tc=2*np.pi/wc
tab_t=np.linspace(0,50*Tc,3000)
ci=[0,v0,0,0,0,v1]
sol=odeint(derivee,ci,tab_t)
tab_x238=sol[:,0]
tab_y238=sol[:,2]
tab_z238=sol[:,4]
m=m235
wc=wc235
sol=odeint(derivee,ci,tab_t)
tab_x235=sol[:,0]
tab_y235=sol[:,2]
tab_z235=sol[:,4]
from mpl_toolkits.mplot3d import Axes3D
fig=plt.figure(figsize=(10,10))
ax = fig.add_subplot(111, projection='3d')
ax.plot(tab_x235,tab_y235,tab_z235,label="U235")
ax.plot(tab_x238,tab_y238,tab_z238,label="U238")
plt.legend()
plt.show()
```



Ça ne change pas grand chose.

[ ]: