

```

import numpy as np

import csv

from scipy.optimize import curve_fit

from scipy.constants import mu_0

import matplotlib.pyplot as plt

import math

"""

e=float(input("épaisseur du rail (en mm)="))/1000

l=float(input("largeur du projectile (en mm)="))/1000

h=float(input("hauteur du rail(en mm)="))/1000

U0=float(input("tension aux bornes du condensateur à t0 (en V)="))

C=float(input("capacité du condensateur (en microF)="))/1000000

R=float(input("résistance du circuit (ohm)="))

"""

e = 6*10**(-3) #in m

l = 6*10**(-3) #in m

h = 6*10**(-3) #in m

U0 = 128-22.5 #on peut prendre la valeur issue du fit

C = 800*10**(-6) #in F

R = 0.2 #on peut prendre la valeur issue du fit

t=0

dt=0.0000001

tho=R*C

s=e*h

mu0=1.2566e-6

```

```
temps=[]
```

```
champ=[]
```

```
force=[]
```

```
vitesse=[]
```

```
joule=[]
```

```
tension=[]
```

```
for i in range (5000):
```

```
    U=U0*math.exp(-t/tho)
```

```
    I=U/R
```

```
    #I=C*U0*math.exp(-t/tho)*(-1/tho)
```

```
    #calcul du champ
```

```
    B=((mu0*I)/(4*I))*(math.log((e/2+I)/(e/2)))
```

```
    F=B*I*I
```

```
    dv=F*dt/0.003
```

```
    P=(U*I)*dt
```

```
    temps.append(t*1000)
```

```
    champ.append(B)
```

```
    force.append(F)
```

```
    vitesse.append(dv)
```

```
    joule.append(P)
```

```
    tension.append(U+22.5)
```

```
    t=t+dt
```

```
E=(C*U0**2)/2
```

```
#intégration des variations de vitesse sur t
```

```
V=sum(vitesse)
```

```
J=sum(joule)
```

```
print("La vitesse est de " + str(V) + " m/s." + "\n")
```

```
    "L'énergie initiale est de " + str(E) + " J." + "\n"
```

```
    "L'énergie perdue par effet Joule est de " + str(J) + " J." + "\n"
```

```
    "L'énergie cinétique est de " + str(0.5*0.003*V**2) + " J." + "\n")
```

```
#affichage des graphes
```

```
def conversion_B(voltage):
```

```
    """
```

```
    Returns the magnetic field, in Tesla, from the output voltage of the sensor
```

```
    10**-4 to convert from G to Tesla
```

```
    """
```

```
    return (((voltage)-2.5)*666)*10**-4)-0.002
```

```
def read_csv(filename):
```

```
    """
```

```
    Fonction pour lire un fichier csv extrait de l'oscillo,
```

```
    temps, voie 1, voie 2
```

```
    """
```

```

#initialisation des arrays qui seront renvoyés par la fonction

data_times = []

data_U = []

data_B = []

with open(filename, newline='') as csvfile:

    spamreader = csv.reader(csvfile, delimiter=',')

    #parcourt les lignes du fichier

    for row in spamreader:

        try: #on essaie de convertir les données de la ligne en nombres

            data_times.append(np.float64(row[0])+2.3*10**-3)

            data_U.append(np.float64(row[1])*1.1)

            data_B.append(np.float64(row[2])/10)

        except: #si premières lignes (pas des chiffres), erreur que l'on ignore

            pass

return np.asarray(data_times), np.asarray(data_U), conversion_B(np.asarray(data_B))

filename = 'graphite 800uF 1er essai.csv'

data_times, data_U, data_B = read_csv(filename) #in s, V, G

plt.figure("B")

plt.plot((data_times[100:]*10**3)-0.04, data_B[100:],label="Courbe expérimentale")

plt.plot(temps,champ,label="Courbe théorique")

```

```
plt.xlabel("Temps (ms)")
plt.ylabel("B (T)")
plt.title("évolution du champ magnétique pour U0=128V et Uf=22.5V R=0,2Ω")
plt.legend()
"""

plt.figure("F")
plt.plot(temps,force)
plt.xlabel("Temps (ms)")
plt.ylabel("F (N)")
plt.title("Evolution de F en fonction de t")
"""

plt.figure("U")
plt.plot((data_times[100:]*10**3)-0.04, data_U[100:],label="Courbe expérimentale")
plt.plot(temps,tension,label="Courbe théorique")
plt.xlabel("Temps (ms)")
plt.ylabel("U (V)")
plt.title("Evolution de la tension pour U0=128V et Uf=22.5V (R=0,2Ω)")
plt.legend()
```