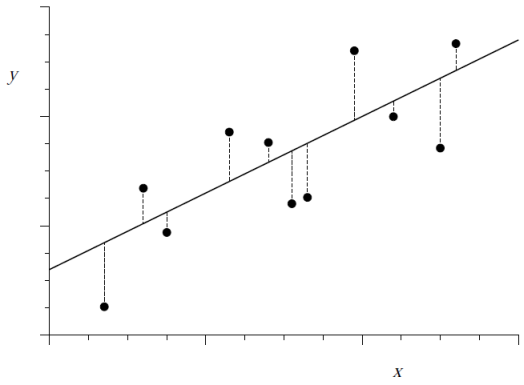
TP 3

Exercice 1 : *Constante de Planck*

Il existe une situation courante en physique, qui se vérifie lorsqu'une expérience produit des données se trouvant approximativement sur une ligne droite, comme les points (ou cercles) de cette figure :



La ligne continue représente ici la ligne droite, que nous ne connaissons pas, et les points représentent les données mesurées se situant à peu près le long de la ligne. La ligne droite peut être représentée sous la forme $y = mx + c$.

...

1. Voici les coordonnées x et y d'un ensemble de valeurs :

5.4874e+14	0.5309
6.931e+14	1.0842
7.4307e+14	1.2734
8.2193e+14	1.6598
9.6074e+14	2.19856
1.184e+15	3.10891

2. Écrivez un programme permettant de lire ces valeurs et tracer un plot avec une valeur ou un cercle pour chaque valeur.
3. Ajoutez dans votre programme le calcul de E_x, E_y, E_{xx} et E_{xy}, puis imprimez les valeurs de m et c.
4. Évaluez $mx_i + c$ pour chaque donnée et tracez la ligne ajustée sur le même graphique.
5. Utilisez les résultats pour calculer la constante de Planck à partir des données de Millikan.

Solution :

```
from numpy import empty
from pylab import plot,show,scatter

x=[5.4874e+14, 6.931e+14, 7.4307e+14, 8.2193e+14, 9.6074e+14, 1.184e+15]
y=[0.5309, 1.0842, 1.2734, 1.6598, 2.19856, 3.10891]
scatter(x,y) # the plot is at the bottom end

N=len(x)
```

```
Ex=0
Ey=0
Exx=0
Exy=0

for i in range(N):
    Ex += x[i]/N
    Ey += y[i]/N
    Exx += x[i]*x[i]/N
    Exy += x[i]*y[i]/N


m = (Exy-Ex*Ey)/(Exx-Ex**2)
c = (Exx*Ey-Ex*Exy)/(Exx-Ex**2)

print ("Coeffients: ",m,c)

yfit = empty(N,float)
for i in range(N):
    yfit[i]= m*x[i]+c
plot(x,yfit)
show()

# to calculate the value of h
e = 1.602E-19
h = m*e
print ("h= ",h,"Js")
```

Exercice 2 : *Volumes*

1. Définir une fonction cube qui calcule le volume d'un cube.
2. Définir une fonction sphere qui calcule le volume d'une sphère en appelant la fonction cube.
3. Définir une fonction à deux arguments qui calcule le volume d'un cône, puis d'une pyramide.
4. Écrire une fonction volume qui prend pour argument une longueur et une chaîne de caractères ("cube","sphere","cone","pyramide") et qui appelle la fonction correspondante.

Solution :

```
from math import pi

def cube(a,verbose=True):
    v=pow(a,3)
    if verbose :
        print ("Le volume du cube d'arete ",a ," m est ",v ,"m^3")
    return v

def sphere(r,verbose=True):
    v=cube(r,False)
    v=4.0/3.0*pi*v
    if verbose :
        print ("Le volume de la sphere de rayon ",r ," m est",v ,"m^3")
    return v

def cone(r,h,verbose=True ):
    v =1/3*pi*pow(r,2)*h
    if verbose :
        print ("Le volume du cone est ",v ," m^3 ")
    return v
```

```
def pyramide (a ,h , verbose = True ):
    v=1/3*pow(a,2)*h
    if verbose :
        print ("Le volume de la pyramide est ",v ," m^3")
    return v

def volume(choix,a1,a2=1):
    if choix == "cube":
        v = cube(a1, False)
    elif choix == "sphere":
        v = sphere(a1,False)
    elif choix == "pyramide":
        v = pyramide(a1,a2,False)
    elif choix == " cone ":
        v = cone(a2,a2,False)
    else:
        v =0
        print ("Ce type de volume n est pas defini")
    return v
```

Exercice 3 : *Tracé de fonctions mathématiques*

1. $f(x) = \frac{1}{1-x}$ pour $x \in [-5, 5]$
2. $f(x) = \begin{cases} e^{\sqrt{1-ax^2}}, & |x| < \sqrt{1/a} \\ 0, & |x| > \sqrt{1/a} \end{cases}$ pour $a = 1, 0.5, 0.1, 0.01$
3. Convertir un angle en radians en degrés

Solution :

```
from math import sqrt , exp
from matplotlib import pyplot

def func1(x):
    f=1/(1-x)
    return f

def func2(x,a=1):
    if abs(x)<sqrt(1/a):
        f=exp(sqrt(1-a*x*x))
    else:
        f=0
    return f


xmin=-5.0
xmax=5.0
npoints=503
pas=(xmax-xmin)/npoints
x=[xmin+i*pas for i in range(npoints+1)]
y=[func1(xx) for xx in x]
y2=[func2(xx,1) for xx in x]
y3=[func2(xx,0.5) for xx in x]
y4=[func2(xx,0.1) for xx in x]

fig = pyplot.figure(1)
pyplot.plot(x,y,color='b')
#pyplot.show() # si on ajoute cette ligne,
```

```
#il faut fermer la figure 1 pour voir la figure 2

fig = pyplot.figure(2)
pyplot.plot(x,y2,color='b')
pyplot.plot(x,y3,color='b')
pyplot.plot(x,y4,color='b')
pyplot.show()
```