Coefficient de trainée d'un ballon de rugby

Afin de réaliser l'étude expérimentale de son mouvement de chute dans l'air, un ballon de rugby est abandonné sans vitesse initiale du haut d'un château d'eau dont la hauteur est 50 m. Sa position au cours du mouvement a été pointée en utilisant un logiciel adapté (par exemple « Latispro »)

Question simple :

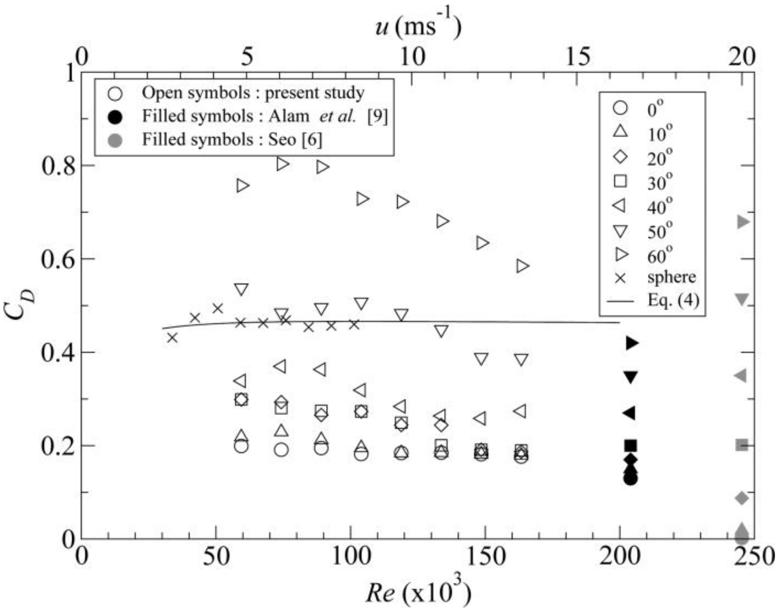
- Décrire le protocole permettant de réaliser l'acquisition de la position et de la vitesse du ballon au cours du mouvement.
- Enoncer la seconde loi de Newton. Etablir l'équation différentielle vérifiée par la composante du vecteur vitesse $\vec{v}$ du ballon dans un référentiel galiléen d'axe Oz ascendant, dans l'hypothèse où le ballon est soumis à une force de frottement de l'air du type $\vec{F} = -\alpha\vec{v}$ (α constante positive)

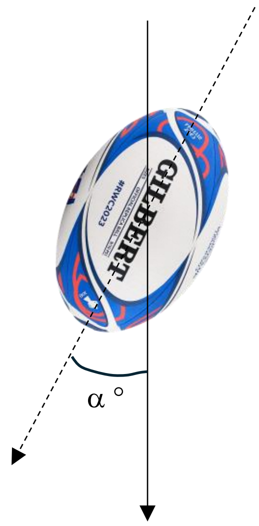
Question ouverte :

Déterminer le coefficient de traînée C_D du ballon de rugby et analyser la modélisation du mouvement à l'aide du script Python dont on complètera les lignes 26, 35 et 40.

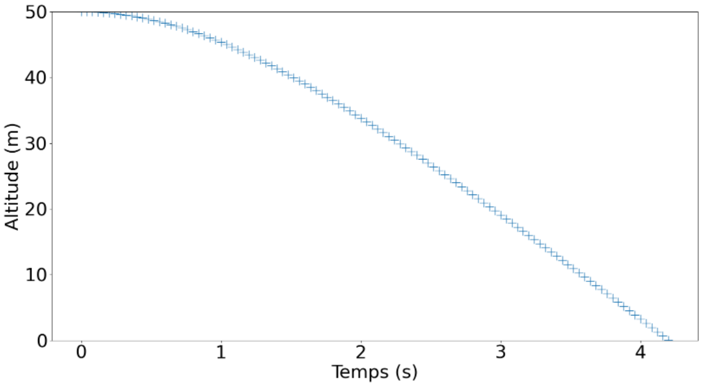
Document 1 : courbe de traînée du ballon de rugby.

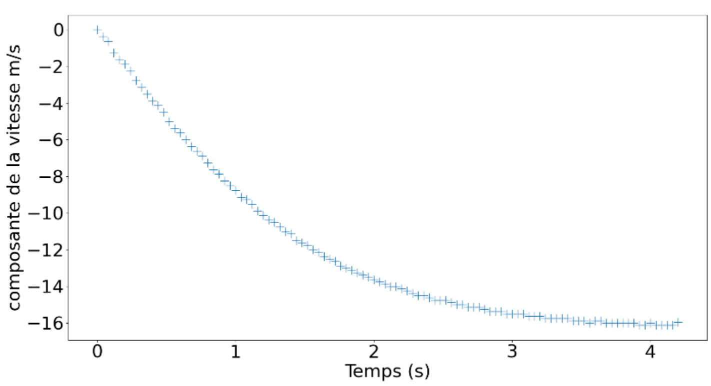
La courbe empirique du coefficient de trainée C_D en fonction du nombre de Reynolds R_e a été obtenue en laboratoire par plusieurs équipes de chercheurs à partir de l'étude du mouvement du ballon dans l'air, pour différents angles α (en °) mesurés entre le grand axe de symétrie du ballon et l'axe du mouvement. Le coefficient de trainée de la sphère est rappelé sur le nuage de points.



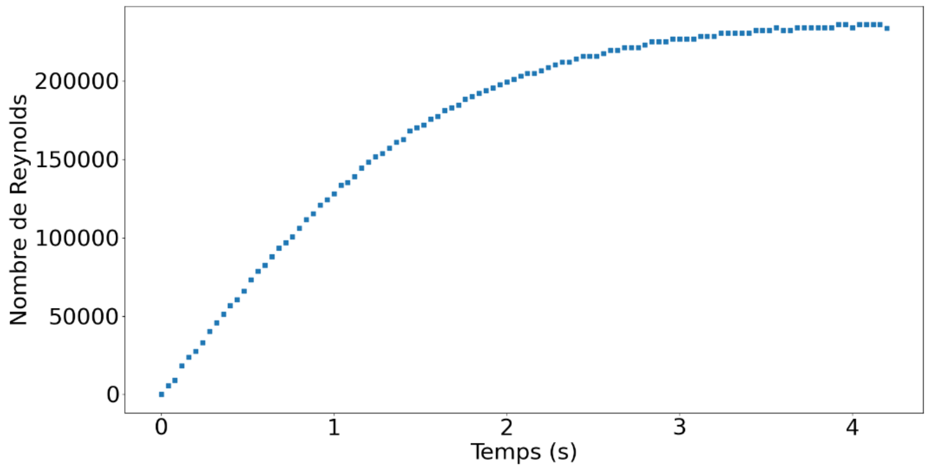


Document 2 : Nuages de points du relevé de l'altitude et de la composante de la vitesse du ballon en fonction du temps.

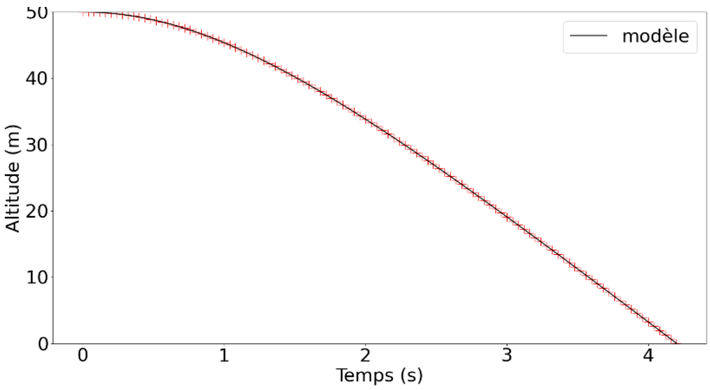


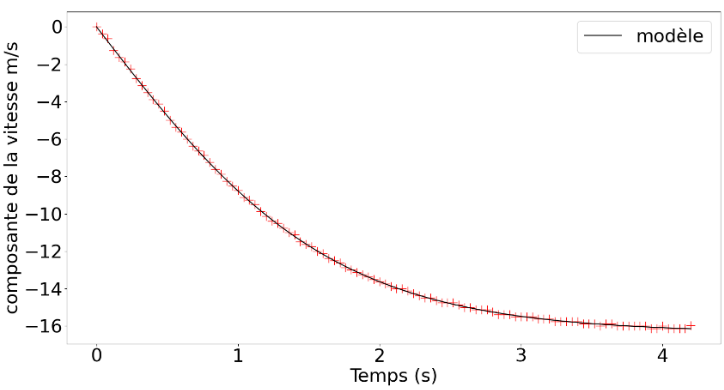


<u>Document 3</u> : Evolution du nombre de Reynolds au cours du mouvement.



<u>Document 4</u> : Résultats de la modélisation renvoyés par le script.





<u>Document 5</u> : script Python

```
# TRAINEE BALLON DE RUGBY_(à  compléter.py

import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
from math import pi


#Données numériques
g=9.8        #accélération de la pesanteur en m/s2
m=0.460      #masse du ballon en kg
D=0.25       #`diamètre moyen de la surface frontale
S=0.06       #Surface frontale du ballon
rho=1.17     #masse volumique de l'air en kg/m^3
eta=2e-5     #viscosité dynamique de l'air en Pl
z0=50        #altitude initiale en m
zpoint0=-0.000001   #composante de la vitesse initiale en m/s

#Relevé des points expérimentaux
#tableau des instants
T=np.array([0.0, 0.04, 0.08, 0.12, 0.16, 0.2, 0.24, 0.28, 0.32, 0.36, 0.4, 0.44, 0.48, 0.52,
0.56, 0.6, 0.64, 0.68, 0.72, 0.76, 0.8, 0.84, 0.88, 0.92, 0.96, 1.0, 1.04, 1.08, 1.12, 1.16,
1.2, 1.24, 1.28, 1.32, 1.36, 1.4, 1.44, 1.48, 1.52, 1.56, 1.6, 1.64, 1.68, 1.72, 1.76, 1.8,
1.84, 1.88, 1.92, 1.96, 2.0, 2.04, 2.08, 2.12, 2.16, 2.2, 2.24, 2.28, 2.32, 2.36, 2.4, 2.44,
2.48, 2.52, 2.56, 2.6, 2.64, 2.68, 2.72, 2.76, 2.8, 2.84, 2.88, 2.92, 2.96, 3.0, 3.04, 3.08,
3.12, 3.16, 3.2, 3.24, 3.28, 3.32, 3.36, 3.4, 3.44, 3.48, 3.52, 3.56, 3.6, 3.64, 3.68, 3.72,
3.76, 3.8, 3.84, 3.88, 3.92, 3.96, 4.0, 4.04, 4.08, 4.12, 4.16, 4.2])
#Valeurs de l'altitude du ballon en m
Z=np.array([50.0, 49.98, 49.97, 49.93, 49.87, 49.8, 49.72, 49.62, 49.5, 49.37, 49.22, 49.06,
48.89, 48.7, 48.49, 48.27, 48.04, 47.79, 47.53, 47.26, 46.98, 46.68, 46.37, 46.05, 45.71, 45.37,
45.01, 44.64, 44.27, 43.88, 43.48, 43.07, 42.65, 42.23, 41.79, 41.35, 40.9, 40.43, 39.97, 39.49,
39.01, 38.52, 38.02, 37.52, 37.01, 36.49, 35.97, 35.44, 34.91, 34.37, 33.83, 33.28, 32.73,
32.17, 31.61, 31.05, 30.48, 29.91, 29.33, 28.75, 28.17, 27.58, 26.99, 26.4, 25.81, 25.21, 24.61,
24.01, 23.4, 22.8, 22.19, 21.58, 20.96, 20.35, 19.73, 19.11, 18.49, 17.87, 17.25, 16.62, 16.0,
15.37, 14.74, 14.11, 13.48, 12.85, 12.22, 11.58, 10.95, 10.31, 9.67, 9.04, 8.4, 7.76, 7.12,
6.48, 5.84, 5.2, 4.56, 3.91, 3.27, 2.63, 1.98, 1.34, 0.69, 0.05])
#Valeurs des composantes de la vitesse du ballon en m/s
ZPOINT=np.array([0, -0.375, -0.625, -1.25, -1.625, -1.875, -2.25, -2.75, -3.125, -3.5,
-3.875, -4.125, -4.5, -5.0, -5.375, -5.625, -6.0, -6.375, -6.625, -6.875, -7.25, -7.625, -7.875,
-8.25, -8.5, -8.75, -9.125, -9.25, -9.5, -9.875, -10.125, -10.375, -10.5, -10.75, -11.0,
-11.125, -11.5, -11.625, -11.75, -12.0, -12.125, -12.375, -12.5, -12.625, -12.875, -13.0,
-13.125, -13.25, -13.375, -13.5, -13.625, -13.75, -13.875, -14.0, -14.0, -14.125, -14.25,
-14.375, -14.5, -14.5, -14.625, -14.75, -14.75, -14.75, -14.875, -15.0, -15.0, -15.125, -15.125,
-15.125, -15.25, -15.375, -15.375, -15.375, -15.5, -15.5, -15.5, -15.5, -15.625, -15.625,
-15.625, -15.75, -15.75, -15.75, -15.75, -15.75, -15.875, -15.875, -15.875, -16.0, -15.875,
-15.875, -16.0, -16.0, -16.0, -16.0, -16.0, -16.0, -16.125, -16.125, -16.0, -16.125, -16.125,
-16.125, -16.125, -15.96])

#Calcul du nombre de Reynolds
#A COMPLETER

#MODELISATION
t0=0  # instant initial(s)
tf=4.4  #  durée (s)
N=111  # nombre de points

def f(U,t):
    (z,zpoint)=U
    #A COMPLETER
    return (zpoint,-g+(Fz/m))

SOLUTION=odeint(f,(z0,zpoint0),T)
Z_mod=SOLUTION[:,0]
#A COMPLETER

plt.figure(1) #tracé du nuage de points des altitudes et du modèle
plt.ylim(0,50)
plt.plot(T,Z,'+',c='r',markersize=15)
plt.plot(T,Z_mod,'k',label='modèle')
plt.xlabel('Temps (s)',fontsize=30)
plt.ylabel('Altitude (m)',fontsize=30)
plt.xticks(fontsize=30)
plt.yticks(fontsize=30)
plt.legend(fontsize=30)
plt.figure(2)#tracé du nuage de points des vitesses et du modèle
plt.plot(T,ZPOINT,'+',c='r',markersize=15)
plt.plot(T,ZPOINT_mod,'k',label='modèle')
plt.xlabel('Temps (s)',fontsize=30)
plt.ylabel('composante de la vitesse m/s',fontsize=30)
plt.xticks(fontsize=30)
plt.yticks(fontsize=30)
plt.legend(fontsize=30)

plt.figure(3) #tracé du nombre de Reynolds
plt.plot(T,RE,'s')
plt.xlabel('Temps (s)',fontsize=30)
plt.ylabel('Nombre de Reynolds',fontsize=30)
plt.xticks(fontsize=30)
plt.yticks(fontsize=30)
plt.show()
```

Annexe : commande Python ODEINT

Cette commande permet de résoudre tout type d'équation différentielle mise sous la forme :

$$\frac{d\boldsymbol{Y}}{dt} = f(\boldsymbol{Y}, t)$$

(ou un système d'équations différentielles, Y est alors un vecteur à plusieurs composantes).

Trois arguments doivent être saisis dans la commande *odeint* pour que l'intégration soit réalisée :

- **odeint (f, Y_0, t)** où Y_0 est la condition initiale (ou les conditions initiales qui sont alors les composantes du vecteur Y_0) et t un tableau contenant les valeurs du temps choisies pour la résolution.
- Les solutions sont renvoyées sous forme d'un tableau, rangées dans le même ordre que les composantes du vecteur Y.