DM21 – Falling faster than g
Correction

Exercice 1 – Qui tombe le plus vite ? – Résolution de problème

1. On commence par déterminer la durée de la chute de la bille. En négligeant les frottements, la bille est en chute libre : le PFD permet d'établir l'équation du mouvement qui, intégrée deux fois donne l'évolution temporelle de l'altitude $y(t)$. On trouve

$$y(t) = h - \frac{g}{2}t^2.$$

La durée τ_{bille} de la chute est telle que $y(\tau_{\text{bille}}) = 0$, soit, en gardant la racine positive :

$$\boxed{\tau_{\text{bille}} = \sqrt{\frac{2h}{g}} = \sqrt{\frac{\ell}{g}},} \quad \text{car} \quad h = \ell \sin\theta_0.$$

La durée τ_{tige} de la chute de la tige s'obtient en reprenant le raisonnement de l'exercice sur la chute d'un arbre du TD M7, à la différence près que l'angle θ est cette fois repéré par rapport à l'horizontale. On obtient, tous calculs faits,

$$\boxed{\tau_{\text{tige}} = \mathcal{I}\sqrt{\frac{\ell}{3g}}.}$$

La comparaison des deux résultats permet de conclure :

$$\boxed{\frac{\tau_{\text{bille}}}{\tau_{\text{tige}}} = \frac{\sqrt{3}}{\mathcal{I}} \approx 1{,}1.}$$

On a donc $\tau_{\text{bille}} > \tau_{\text{tige}}$: **l'extrémité A de la tige touchera le sol avant la bille.**

2. **Ce résultat ne peut se maintenir** quelle que soit la position angulaire initiale de la tige. En effet dans le cas où la tige est initialement quasi verticale, son temps de chute devient très grand et tend même vers l'infini pour $\theta_0 = \pi$, alors que le temps de chute de la bille vaut au maximum $\sqrt{2\ell/g}$. Il existe donc une valeur critique θ_c de θ_0, telle que si $\theta_0 > \theta_c$, la bille tombe avant la tige.

Numériquement, on peut estimer la valeur critique de l'angle initial pour laquelle les temps de chute sont égaux.

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import quad
from scipy.optimize import bisect

################################
# PARAMÈTRES DE LA SIMULATION #
################################
l = 1               # longueur de la tige (m)
g = 9.81            # accélération de la pesanteur (m/s^2)
```

```
theta0 = np.pi/6  # inclinaison initiale de la tige (rad)

##############################
# CALCUL DES TEMPS DE CHUTE #
##############################
def tau_bille(theta0):
    """Temps de chute de la bille (s)"""
    return np.sqrt( 2 * l * np.sin(theta0) / g )

def tau_tige(theta0):
    """Temps de chute de la tige (s)"""
    def f(theta):
        return 1/np.sqrt( np.sin(theta0) - np.sin(theta) )
    I = quad(f,0,theta0)[0]
    return I * np.sqrt(l / 3 / g)

############################
# AFICHAGE DES RÉSULTATS #
############################
print("Pour un angle initial de 30°, les temps de chute sont")
print("tau_bille          : {:.0f} ms".format(tau_bille(theta0)*1e3))
print("tau_tige           : {:.0f} ms".format(tau_tige(theta0)*1e3))
print("tau_bille/tau_tige : {:.2f}".format(tau_bille(theta0)/tau_tige(theta0)))

############################
# REPRÉSENTATION GRAPHIQUE #
############################
plt.figure()
plt.clf()
theta0 = np.linspace(1, 89, 89)
plt.xlim(0,90)
plt.ylim(0,1)
plt.grid()
plt.xlabel("Angle initial $\\theta_0$ (°)")
plt.ylabel("Temps de chute (s)")
plt.plot(theta0, tau_bille(theta0*np.pi/180), label="bille")
plt.plot(theta0, [tau_tige(a*np.pi/180) for a in theta0], label="tige")
plt.legend()
plt.show()

##################################
# ESTIMATION DE L'ANGLE CRITIQUE #
##################################
def ecart(theta0):
    """Différence entre temps de chute"""
    theta0 *= np.pi/180
    return tau_tige(theta0) - tau_bille(theta0)

theta_c = bisect(ecart, 1, 89)
print("Angle critique : {:.1f}°".format(theta_c))
```