

1 7b9a-1603552

## Mouvement d'un Punching-Ball

On s'intéresse ici à l'étude du mouvement d'un punching-ball ci-contre, utilisé lors des entraînements des boxeurs. Le mouvement du punching-ball peut-être décrit à l'aide de l'angle  $\theta$  que forme la barre métallique reliant le socle immobile à la zone frappée, par rapport à la verticale. Ce punching-ball constitue un pendule inversé, maintenu autour de sa position d'équilibre verticale grâce à un couple de rappel proportionnel à l'angle  $\theta$ . Il subit en outre un couple de frottement fluide proportionnel à sa vitesse de rotation.

Lorsqu'un boxeur frappe la partie supérieure du punching-ball, celui-ci oscille plusieurs fois avant de se stabiliser à nouveau.

**Q1.** En introduisant les grandeurs et hypothèses nécessaires, montrer que l'angle  $\theta$  est soumis, juste après la frappe du joueur, à une équation du type :

$$\frac{d^2\theta}{dt^2} + b\frac{d\theta}{dt} + c\theta = d \sin(\theta)$$

où on exprimera  $b, c$  et  $d$  en fonction des paramètres introduits.

La résolution de cette équation nécessite de définir des conditions initiales.

**Q2.** Proposer ainsi des conditions initiales cohérentes avec la situation physique étudiée.

**Q3.** Montrer que la position d'équilibre initiale du pendule n'est stable que si le coefficient de rappel  $k$  est supérieur à une certaine valeur dont on donnera l'expression.



### Donnée :

Si le mouvement d'un système autour d'une position d'équilibre est régi par une équation du type :

$$\alpha f'''(t) + \beta f''(t) + \gamma f'(t) = \delta$$

(où  $\alpha, \beta, \gamma$  et  $\delta$  sont des constantes réelles)

Alors, pour que la position d'équilibre de ce système soit stable, il faut que  $\alpha, \beta$  et  $\gamma$  soient de même signe.

On donne les fonctions suivantes qu'il conviendra d'utiliser sans les modifier par la suite.



In [2]:

```

1  from math import*
2  import numpy as np
3  import matplotlib.pyplot as plt
4
5  def F(theta,thetapoint):
6      h=1
7      k=30
8      m=1
9      l=1.5
10     J=m*l*l
11     g=10
12     x=m*g*l/J*sin(theta)-k/J*theta-h/J*thetapoint
13     return thetapoint,x
14
15
16
17 def F2(theta,thetapoint):
18     h=1
19     k=13
20     m=1
21     l=1.5
22     J=m*l*l
23     g=10
24     x=m*g*l/J*sin(theta)-k/J*theta-h/J*thetapoint
25     return thetapoint,x
26
27 def Euler(F,Texp,dt,v):
28     l=1.5
29     theta=[0]
30     thetapoint=[v/l]
31     t=[0]
32     N=int(Texp/dt)
33     for i in range(N-1):
34         a,b=F(theta[-1],thetapoint[-1])
35         if theta[-1]+a*dt>3.14/2 or theta[-1]+a*dt<-3.14/2 :
36             theta.append(theta[-1])
37             thetapoint.append(-thetapoint[-1])
38             t.append((i+1)*dt)
39         else :
40             theta.append(theta[-1]+a*dt)
41             thetapoint.append(thetapoint[-1]+b*dt)
42             t.append((i+1)*dt)
43     return t,theta,thetapoint
44
45 def portrait(v):
46     t,theta,thetapoint=Euler(F,25,0.001,v)
47     plt.close()
48     plt.figure()
49     plt.plot(theta,thetapoint)
50     plt.xlabel("Theta (rad)")
51     plt.ylabel("d Theta / dt (Rad/s)")
52     plt.title("Portrait de phase")
53     plt.grid()
54     plt.show()
55
56 def portrait2(v):
57     t,theta,thetapoint=Euler(F2,25,0.001,v)
58     plt.close()
59     plt.figure()

```

```
60 plt.plot(theta, thetapoint)
61 plt.xlabel("Theta (rad)")
62 plt.ylabel("d Theta / dt (Rad/s)")
63 plt.title("Portrait de phase")
64 plt.grid()
65 plt.show()
66
```

**Q4** Expliquer brièvement la méthode de Cauchy pour résoudre des équations différentielles de degré  $n = 2$ .

**Q5** Apporter des "doc string" à chaque fonction afin de mieux apprécier leur intérêt.

**Q6** Tracer les portraits dans différentes situations. Commenter