

```

0  ### -*- coding: cp1252 -*-
1  """
2  Étude de l'oscillateur de Duffing
3  Calcul de la période
4  """
5  from __future__ import division # pour éviter la division entière
6
7  from pylab import *
8  from scipy import *
9  from scipy.integrate import odeint
10
11 #initialisation
12 omega_ini = 0
13 tini = 0
14 tfin = 10
15 Npas = 10000.
16 precis = (tfin - tini)/Npas
17 print 'precision = ', precis
18
19 precision = 0.001
20
21 omega0 = 1
22 om2 = omega0**2
23 NbrePoints = 30
24 theta_ini_first = 0.1
25 theta_ini_last = 1
26 theta_ini_tab = linspace(theta_ini_first, theta_ini_last, NbrePoints)
27 Tm_tab = array([])
28
29 def Fduffing(Y, t):
30     [theta, omega] = Y
31     eq1 = omega
32     eq2 = -om2*theta*(1-theta**2/6)
33     return [eq1, eq2]
34
35 t = linspace(tini, tfin, Npas)
36
37 for theta_ini in theta_ini_tab:
38     cond_ini = [theta_ini, omega_ini]
39     Yn = odeint(Fduffing, cond_ini, t)
40     [theta, omega] = Yn.T
41
42     Ne = size(theta) # détermination du nombre d'éléments dans theta
43
44     T_pass_zero = array([])
45     for n in range(Ne-1):
46         if (theta[n]*theta[n+1]<0) and (omega[n]<0):
47             T_pass_zero = append(T_pass_zero, t[n])
48
49     #print T_pass_zero
50     Npz = size(T_pass_zero) # détermination du nombre d'éléments dans
51                             T_pass_zero
52
53     Tm = 0
54     for n in range(1, Npz):
55         Tm = Tm + T_pass_zero[n]-T_pass_zero[n-1]
56     Tm = Tm/(Npz-1)

```

```

53 # print 'Tm =', Tm
54 Tm_tab = append(Tm_tab, Tm)
55
56 plot(theta_ini_tab**2, Tm_tab, '-o', label=ur'Solution numérique')
57
58 theta_ini_th = linspace(theta_ini_first, theta_ini_last, 100)
59 T_th = 2*pi/omega0*(1+theta_ini_th**2/16)
60 plot(theta_ini_th**2, T_th, color='r', label=ur'Solution de Borda')
61 legend()
62
63 show()

```

