

```

0  ### -*- coding: cp1252 -*-
1  """
2  Étude de l'oscillateur harmonique sans dissipation
3  """
4  from __future__ import division # pour éviter la division entière
5
6  from pylab import *
7  from scipy import *
8  from scipy.integrate import odeint
9
10 #initialisation
11 theta_ini = 1
12 omega_ini = 0
13 tini = 0
14 tfini = 50
15 Npas = 400
16 omega0 = 1
17 om2 = omega0**2
18
19 def FL(Y, t, lam):
20     [theta, omega] = Y
21     eq1 = omega
22     eq2 = -om2*theta - 2*lam*omega
23     return [eq1, eq2]
24
25 def FNL(Y, t, lam):
26     [theta, omega] = Y
27     eq1 = omega
28     eq2 = -om2*theta - 2*lam*omega*abs(omega)
29     return [eq1, eq2]
30
31 def FNL2(Y, t, lamL, lamNL):
32     [theta, omega] = Y
33     eq1 = omega
34     eq2 = -om2*theta - 2*lamL*omega - 2*lamNL*omega*abs(omega)
35     return [eq1, eq2]
36
37 cond_ini = [theta_ini, omega_ini]
38 t = linspace(tini, tfini, Npas)
39 # régime linéaire
40 lam1 = omega0*0.2
41 Yn = odeint(FL, cond_ini, t, args=(lam1,))
42 [theta, omega] = Yn.T
43 plot(t, theta, '-b', color='b', label=ur'linéaire')
44 # régime non-linéaire
45 lam2 = omega0*1
46 Yn = odeint(FNL, cond_ini, t, args=(lam2,))
47 [theta, omega] = Yn.T
48 plot(t, theta, '-g', color='g', label=ur'non-linéaire')
49 # régime linéaire + non-linéaire
50 Yn = odeint(FNL2, cond_ini, t, args=(lam1, lam2))
51 [theta, omega] = Yn.T
52 plot(t, theta, '-r', color='r', label=ur'les deux')
53
54 legend()
55

```

56 show()

