Actividad 5: Periodo del péndulo

Valenzuela Chaparro Hugo de Jesús

7 de mayo de 2016

1. Ecuaciones de movimiento del péndulo

El movimiento del péndulo se puede representar por medio de la siguiente ecuación diferencial:
$\frac{d^2\theta}{dt^2} + \frac{g}{\ell} \sin\theta = 0$ ...... (Eq. 1)
donde g es la aceleración debida a la fuerza gravitacional, ℓ la longitud del péndulo y θ el desplazamiento angular respecto a la vertical.

A esta ecuación se le puede solucionar fácilmente usando la aproximación para angulos pequeños cuando $\sin\theta \approx \theta$, que se reduce a la de un oscilador armónico simple. Sin embargo esto no cubriría todo el modelo físico.

2. Solución de la ecuación usando Python

Podemos solucionar la ecuación usando la función scipy.integrate.odeint de Python. Esta función sirve para ecuaciones diferenciales de primer orden, así que primero debemos convertir nuestra ecuación del péndulo en una de primer orden definiendo a theta'(t) (velocidad angular) como omega(t) y a theta"(t) como omega'(t):

theta'(t) = omega(t)

omega'(t) = -b*omega(t) - c*sin(theta(t))

El elemento -b*omega(t) es para cuando actúa fricción en el péndulo, es decir, que es amortiguado, es cuando dicha fuerza es proporcional a la velocidad con el factor b (coeficiente de amortiguamiento) $F = -b\frac{dy}{dt} = -bV$.

3. Código, gráficas y resultados

Se muestran los códigos para resolver la ecuación del péndulo, sin fricción y con fricción, obteniendo así el ángulo a determinado tiempo. En las gráficas se aprecian los ángulos y velocidades en función del tiempo.

- Sin fricción, ángulo inicial de $\pi - 0.1$, sin velocidad inicial y longitud de la cuerda de 1 m.

  Código:

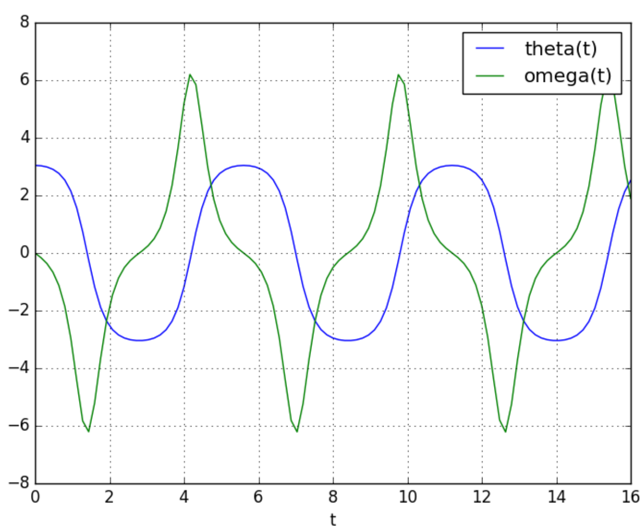
```
from scipy.integrate import odeint
import matplotlib.pyplot as plt
import numpy as np
g=9.8

def pend(y, t, b, c):
        theta, omega = y
        dydt = [omega, - c*np.sin(theta)]
        return dydt

#longitud del péndulo
l=1
c = g/l
#[angulo inicial, velocidad inicial]
y0 = [np.pi-0.1, 0.0]
t = np.linspace(0, 16, 101)

#solucion
sol = odeint(pend, y0, t, args=(b, c))

plt.plot(t, sol[:, 0], 'b', label='theta(t)')
plt.plot(t, sol[:, 1], 'g', label='omega(t)')
plt.legend(loc='best')
plt.xlabel('t')
plt.grid()
plt.show()
```



- Sin fricción, ángulo inicial de $\frac{\pi}{2}$, 5 rad/s velocidad inicial y longitud de la cuerda de 2 m.

  Código:

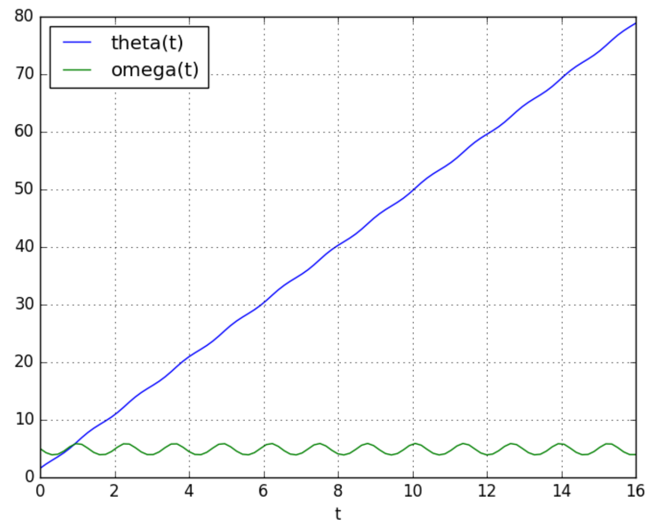
```
from scipy.integrate import odeint
import matplotlib.pyplot as plt
import numpy as np
g=9.8

def pend(y, t, b, c):
        theta, omega = y
        dydt = [omega, - c*np.sin(theta)]
        return dydt

#longitud del péndulo
l=2
c = g/l
#[angulo inicial, velocidad inicial]
y0 = [np.pi/2, 5]
t = np.linspace(0, 16, 101)

#solucion
sol = odeint(pend, y0, t, args=(b, c))

plt.plot(t, sol[:, 0], 'b', label='theta(t)')
plt.plot(t, sol[:, 1], 'g', label='omega(t)')
plt.legend(loc='best')
plt.xlabel('t')
plt.grid()
plt.show()
```



Se observa que como no hay amortiguamiento, y se le aplicó una velocidad inicial, el péndulo irá aumentando el ángulo en el tiempo, es decir, está girando infinitamente.

- Con fricción (coeficiente de amortiguamiento $b = 0.5$), ángulo de $\pi - 0.1$, velocidad inicial 0 y longitud de la cuerda 1m.

  Código:

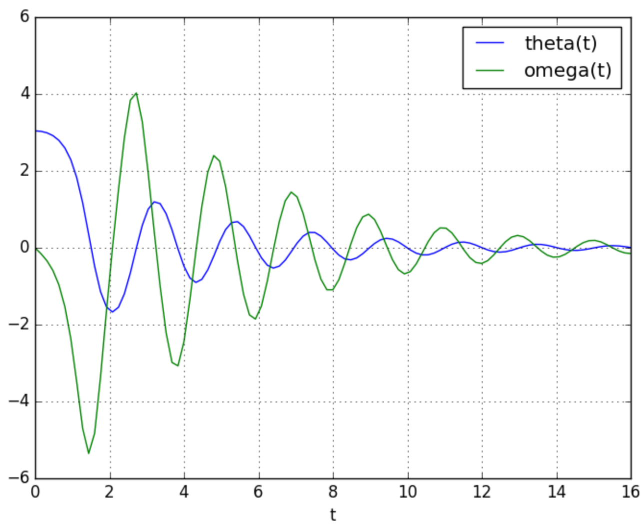
```
from scipy.integrate import odeint
import matplotlib.pyplot as plt
import numpy as np
g=9.8

def pend(y, t, b, c):
        theta, omega = y
        dydt = [omega, -b*omega - c*np.sin(theta)]
        return dydt

#coeficiente de amortiguamiento
b = 0.5
#longitud del péndulo
l=1
c = g/l
#[angulo inicial, velocidad inicial]
y0 = [np.pi-0.1, 0.0]
t = np.linspace(0, 16, 101)
```

```
#solucion
sol = odeint(pend, y0, t, args=(b, c))

plt.plot(t, sol[:, 0], 'b', label='theta(t)')
plt.plot(t, sol[:, 1], 'g', label='omega(t)')
plt.legend(loc='best')
plt.xlabel('t')
plt.grid()
plt.show()
```



Ya hay amortiguamiento, se observa que el péndulo se detendrá

- Con fricción (coeficiente de amortiguamiento $b = 0.5$), ángulo inicial de $\frac{\pi}{2}$, velocidad inicial 5 rad/s y longitud de la cuerda 2m.

  ódigo:

```
from scipy.integrate import odeint
import matplotlib.pyplot as plt
import numpy as np
g=9.8

def pend(y, t, b, c):
        theta, omega = y
        dydt = [omega, -b*omega - c*np.sin(theta)]
        return dydt

#coeficiente de amortiguamiento
```

```
b = 0.5
#longitud del péndulo
l=2.0
c = g/l
#[angulo inicial, velocidad inicial]
y0 = [np.pi/2.0, 5.0]
t = np.linspace(0, 16, 101)

#solucion
sol = odeint(pend, y0, t, args=(b, c))

plt.plot(t, sol[:, 0], 'b', label='theta(t)')
plt.plot(t, sol[:, 1], 'g', label='omega(t)')
plt.legend(loc='best')
plt.xlabel('t')
plt.grid()
plt.show()
```