Actividad 5: Movimiento armónico simple: Péndulo

Jose Pablo Salazar Velazquez

14 de marzo de 2016

1. Introducción

Recordando la primer practica del parcial, analizamos el comportamiento de un pendulo simple y como variaba su periodo cuando se utilizaba una amplitud incial arbitraria en comparacion de solamente angulos pequeños. Haciendo un poco de memoria, podemos recordar que:
La ecuación diferencial que representa el movimiento de un péndulo simple es:

$$\frac{d^2\theta}{dt^2} + \frac{g}{l}\sin\theta = 0 \tag{1}$$

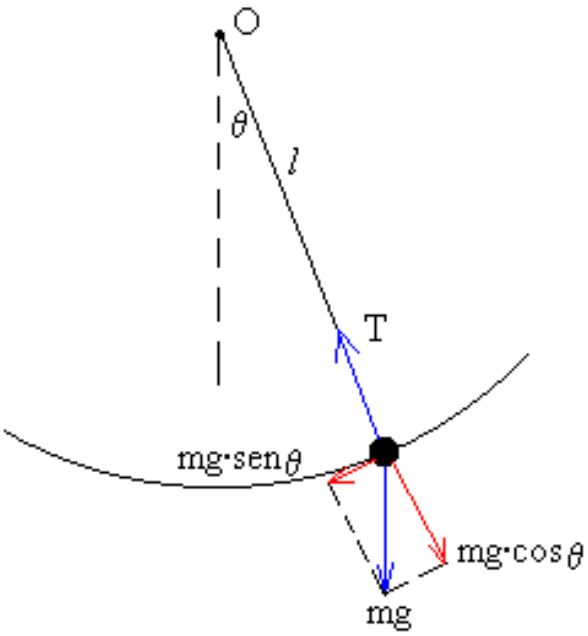
donde g es la aceleración causada por la gravedad, l es la longitud del péndulo y θ es el desplazamiento angular.
La ecuación diferencial que describe el movimiento del péndulo simple no se resuelve sencillamente, y no hay solución analítica para ella. Pero si se añade una restricción en la amplitud de oscilación, resulta una forma de donde es fácil de obtener una solución. Si asumimos que $\theta \ll 1$, entonces $\sin\theta \approx \theta$, obteniendo así la ecuación del oscilador armónico:

$$\frac{d^2}{dt^2} + \frac{g}{l}\theta = 0 \tag{2}$$

Si resolvemos con las siguientes condiciones iniciales, $\theta(0) = \theta_0$, $\frac{d\theta}{dt}(0) = 0$:

$$\theta(t) = \theta_0 \cos\left(\sqrt{\frac{g}{l}}t\right) \qquad \theta_0 \ll 1 \tag{3}$$

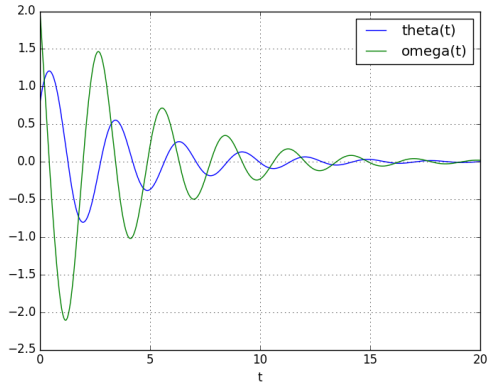
Este resulta en un movimiento armónico simple.



El método *scipy.integrate.odeint* nos ayuda a resolver ecuaciones diferenciales ordinarias al transformarlas en ecuaciones de primer orden utilizando el metodo de cambio de variable. De esta manera, podemos obtener una solución numérica para la ecuación.

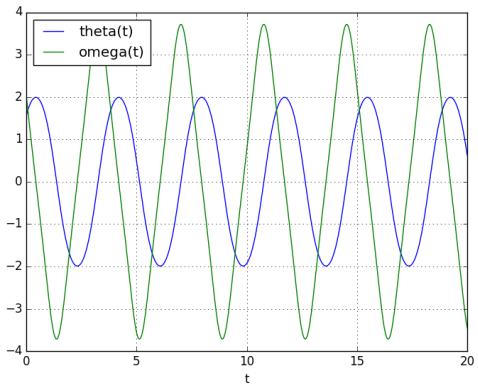
Programa: Ecuación de un péndulo. Se presenta el codigo que se utilizo para resolver la practica 5. Variando los valor de velocidad angular y amplitud del pendelo, asi como el coeficiente de amortiguamiento, podremos ver en las graficas como va variando teta con respecto del tiempo. Podemos ver como varia cuando cambiamos los parametros iniciales. El método *scipy.integrate.odeint* es una herramienta que facilita el resolver E.D.O. al transformarlas en ecuaciones de primer orden por medio de un cambio de variable.

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

#Para definir las ecuaciones
def pend(y, t, b, c):
        theta, omega = y
        dydt = (omega, -b*omega - c*np.sin(theta))
        return dydt

#Propiedades de este pendulo
b = 0.5 #coef de amortiguamiento
g= 9.8 #aceleracion de la gravedad
l=2 #longitud de el pendulo
c=g/l

#Condiciones
y0 = [np.pi/4 , 2 ]

#Intervalo
t = np.linspace(0, 20, 1000)

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

#graficas
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()
```



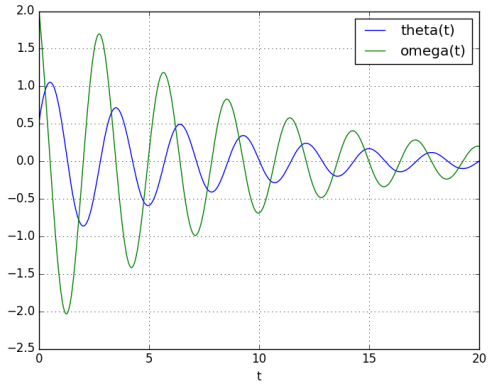

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint

#Para definir las ecuaciones
def pend(y, t, b, c):
        theta, omega = y
        dydt = (omega, -b*omega - c*np.sin(theta))
        return dydt

#Propiedades de este pendulo
b = 0 #coef de amortiguamiento
g= 9.8 #aceleracion de la gravedad
l=2 #longitud de el pendulo
c=g/l

#Condiciones
y0 = [np.pi/2 , 2 ]

#Intervalo
t = np.linspace(0, 20, 1000)

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

#graficas
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()
```




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

#Para definir las ecuaciones
def pend(y, t, b, c):
        theta, omega = y
        dydt = (omega, -b*omega - c*np.sin(theta))
        return dydt

#Propiedades de este pendulo
b = 0.25 #coef de amortiguamiento
g= 9.8 #aceleracion de la gravedad
l=2 #longitud de el pendulo
c=g/l

#Condiciones
y0 = [np.pi/6 , 2 ]

#Intervalo
t = np.linspace(0, 20, 1000)

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

#graficas
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()
```



Conclusión

Al modelar este fenomeno fisico, podemos comprender mejor como es que funciona el amortiguamiento y gracias a estas herramientas, podriamos ver cual es lo opotimo en cada caso

Referencias

[1] Wikipedia, https://es.wikipedia.org/wiki/Pendulo

[2] Scipy, http://scipy.github.io/devdocs/generated/scipy.integrate.odeint.html