Actividad 5

Carolina Valenzuela Córdova

23 de Febrero de 2016

1. Problema del péndulo

El péndulo es un sistema físico que puede oscilar bajo la acción gravitatoria u otra característica física (elasticidad, por ejemplo) y que está configurado por una masa suspendida de un punto o de un eje horizontal fijos mediante un hilo, una varilla, u otro dispositivo que sirve para medir el tiempo.

Existen muy variados tipos de péndulos que, atendiendo a su configuración y usos, reciben los nombres apropiados: péndulo simple, péndulo compuesto, péndulo cicloidal, doble péndulo, péndulo de Foucault, péndulo de Newton, péndulo balístico, péndulo de torsión, péndulo esférico, etcétera.[1]
En general, las matemáticas de los péndulos son muy complicadas. sin embargo, se pueden tomar en cuenta suposiciones que facilitan los cálculos, como en el caso de un péndulo simple, donde las ecuaciones de movimiento puede resolverse de manera analítica para oscilaciones de ángulos pequeños.

2. Ecuación de movimiento y soluciones numéricas

Para esta actividad se nos solicitó un programa que fuera capaz de resolver la ecuación de movimiento del péndulo de forma numérica, pues dicha expresión no tiene las consideraciones antes mencionadas para ángulos pequeños, sino para cualquier ángulo. Deseamos encontrar la posición del péndulo a cualquier tiempo, si conocemos sus condiciones iniciales. La ecuación de movimiento del péndulo está dada por:

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

También se pide probar diferentes casos para mostrar que el código arroja resultados físicos.

3. Elaboración de la actividad

Se nos proporcionó un código ejemplo donde se resuelve una ecuación diferencial de se-

gundo orden para cualquier ángulo theta.
A continuación se presentan los pasos que se siguieron según la referencia[2] para la elaboración del código:

- Primero definimos la ecuación, la cual se escribe como:
  theta"(t)+b*theta'(t)+c*sin(theta(t))=0
- Las tildes utilizadas son para denotar derivadas. Para resolverla, se utlizó la función scipy.integrate.odeint de Python.
- Para poder utilizar dicha función, debemos convertir la ecuación en un sistema de ecuaciones de primer orden. Definimos la velocidad angular omega (t)=theta'(t), y obtenemos el sistema:
  theta'(t)=omega(t)
  omega'(t)= -b*omega(t)-c*sin(theta(t))
- El vector y será [theta,omega], e introducimos este sistema en python como:

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

- Le damos valores iniciales arbitrarios a b y c.
- Para considerar condiciones iniciales, asumimos que el péndulo está vertical con $\theta(0) = \pi - 0.1$, y que está inicialmente en reposo, entonces $\omega(0) = 0$. Así, nuestro vector de condiciones iniciales es:

```
>>> y0 = [np.pi - 0.1, 0.0]
```

- Generamos una solución 101 con muestras uniformemente espaciadas en el intervalo $0 <= t <= 10$, así nuestro arreglo de tiempos es:

```
>>> t = np.linspace(0, 10, 101)
```

- Llamamos a odeint para generar la solución. Le damos los parámetros b y c utilizando $args$:

```
>>> from scipy.integrate import odeint
>>> sol = odeint(pend, y0, t, args=(b, c))
```

- Para graficarlos, necesitamos agregar al código lo siguiente, de tal manera que la primera columna sea $\theta(t)$ y la segunda $\omega(t)$:

```
>>> import matplotlib.pyplot as plt
>>> 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()
```

4. Resultados

4.1. Primer caso: Péndulo sin fricción

En el primer caso, no modificamos nada del código, únicamente se corrió como lo

proponía la página, teniendo como resultados lo siguiente:

```
#theta''(t) + b*theta'(t)
 + c*sin(theta(t)) = 0
#theta'(t) = omega(t)
#omega'(t) = -b*omega(t) -
c*sin(theta(t))
def pend(y, t, b, c):
theta, omega = y
dydt = [omega, -b*omega -
c*np.sin(theta)]
return dydt

b = 0
c = 0.25
y0 = [np.pi - 0.1, 0.0]
t = np.linspace(0, 100, 101)

from scipy.integrate import
odeint
sol = odeint(pend, y0, t,
 args=(b, c))
import matplotlib.pyplot as plt

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()
```

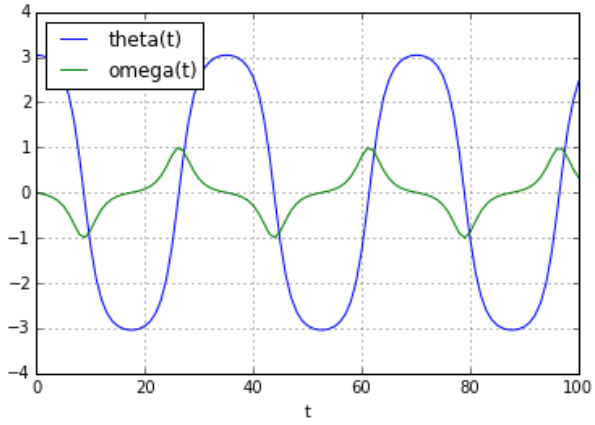

Figura 1: Péndulo sin fricción

4.2. Segundo caso: Péndulo con fricción

En este caso, lo único que se modificó fue que se le dio un valor de 0.5 al parámetro b, el cual indica la fricción en el código, teniendo como resultado:

```
contenidos.#theta''(t) + b*theta'
(t) + c*sin(theta(t)) = 0
#theta'(t) = omega(t)
#omega'(t) = -b*omega(t) - c*sin(theta(t))
def pend(y, t, b, c):
theta, omega = y
dydt = [omega, -b*omega - c*np.sin(theta)]
return dydt

b = 0.5
c = 0.25
y0 = [np.pi - 0.1, 0.0]
t = np.linspace(0, 100, 101)

from scipy.integrate import odeint
sol = odeint(pend, y0, t, args=(b, c))
```

```
import matplotlib.pyplot as plt

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()..
```

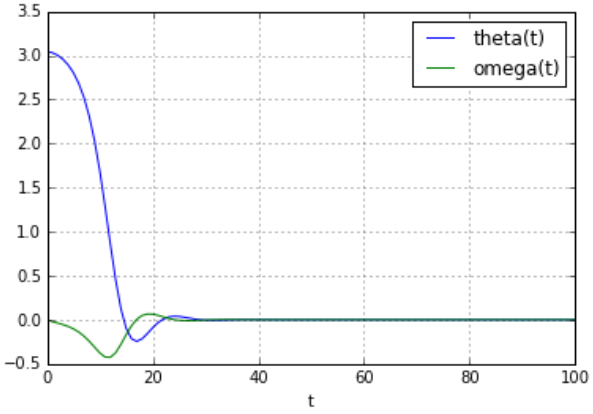

Figura 2: Péndulo con fricción

Aquí se puede osbervar una gran diferencia en las gráficas debido a la naturaleza de lo que estamos modelando, ya que el péndulo pierde energía cinética de manera mucho más rápida a medida que la fricción actúa sobre él.

4.3. Tercer caso: Cambio de ángulo sin fricción

En este caso hicimos un cambio en el ángulo θ para observar qué sucedía con las gráficas que nos arroja el programa. En el código solo se modificó el valor del parámetro $y0$, tomando como ángulo $\theta = \frac{\pi}{6}$, y conservando el mismo valor del parámetro $b = 0$ que se tenía inicialmente.

```
contenidos.#theta''(t) + b*theta'
(t) + c*sin(theta(t)) = 0
#theta'(t) = omega(t)
#omega'(t) = -b*omega(t) - c*sin(theta(t))
def pend(y, t, b, c):
theta, omega = y
dydt = [omega, -b*omega - c*np.sin(theta)]
return dydt

b = 0.5
c = 0.25
y0 = [np.pi/6, 0.0]
t = np.linspace(0, 100, 101)

from scipy.integrate import odeint
sol = odeint(pend, y0, t, args=(b, c))
import matplotlib.pyplot as plt

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()..
```

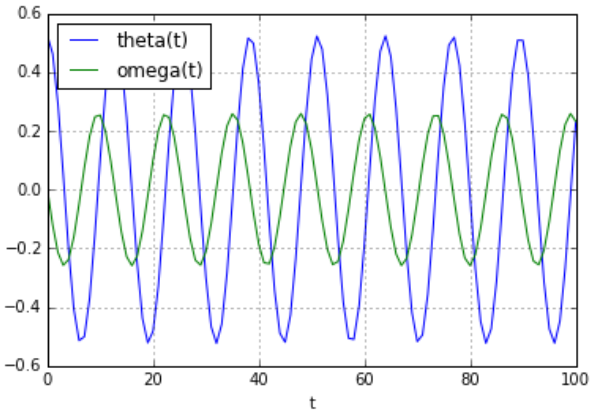

Figura 3: Péndulo sin fricción y $\theta = \frac{\pi}{6}$

Se repitió este proceso conservando el mismo valor de fricción, pero cambiando el ángulo por $\theta = \frac{\pi}{2}$, y obtuvimos como resultado la siguiente gráfica:

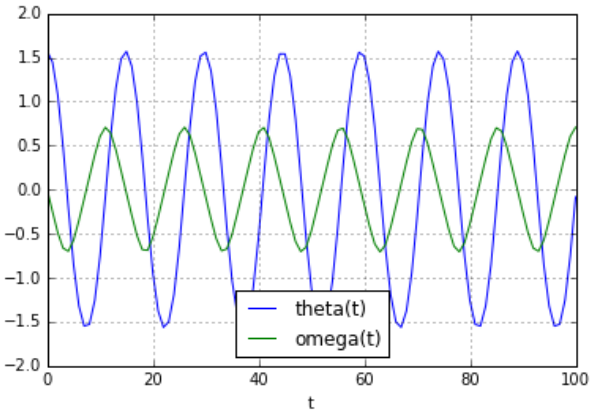

Figura 4: Péndulo sin fricción y $\theta = \frac{\pi}{2}$

4.4. Cuarto caso: Cambio de ángulo con fricción

En este caso, utilizamos el mismo valor de $\theta = \frac{\pi}{2}$ de la última gráfica pero con el parámetro $b = 0.5$ para observar la diferencia entre este caso y el anterior, en los cuales el valor del ángulo coincide. Obtuvimos así la siguiente gráfica:

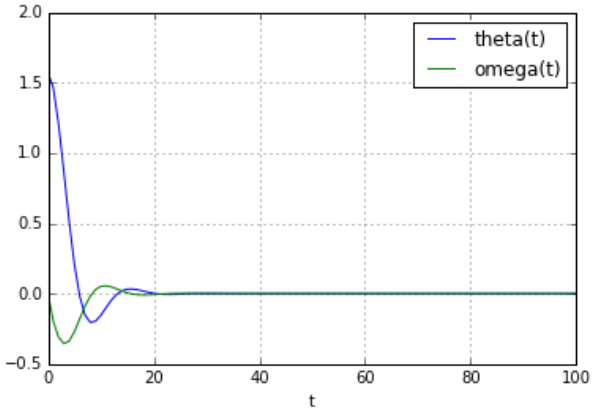

Figura 5: Péndulo con fricción y $\theta = \frac{\pi}{2}$

5. Conclusiones

Fue muy interesante aprender a utilizar herramientas que nos ayuden a modelar fenómenos físicos que se presentan con frecuencia en nuestra vida diaria como estudiantes. Fue además muy útil entender cómo se construye un programa que pueda resolver ecuaciones diferenciales de movimiento presentes en estos modelos matemáticos.

Pudimos notar la congruencia que tenían las gráficas que nos arrojaba el programa con el comportamiento real de un péndulo al ser es-

tudiado.

Referencias

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

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