Actividad 7

Carolina Valenzuela Córdova

3 de Marzo de 2016

1. Teoría

En las actividades pasadas se simuló el comportamiento del péndulo simple, considerando o no la fricción, así como el periodo para ángulos pequeños y arbitrarios.
Hemos estudiado de muchas formas cómo varían los elementos de este modelo matemático, pero en esta actividad construiremos el retrato del Espacio Fase, que representa la familia de soluciones del péndulo simple.
El retrato fase es una representación geométrica de las trayectorias de un sistema dinámico en un plano fase. Cada conjunto de condiciones iniciales es representado por una curva o punto diferente.
Son una herramienta invaluable en el estudio de sistemas dinámicos, como:

- Péndulo simple
- Oscilador armónico simple
- Oscilador de Van der Pol
- Diagrama de bifurcación
- Plano paramétrico

2. Actividad

Para esta actividad se solicitó adaptar un código ejemplo de Lotka-Volterra (el cual aparece en ScyPy CookBook),y construir la figura del retrato fase para una colección de condiciones iniciales. Este representa la familia de soluciones del péndulo simple.
Para conseguir esto, se utilizó la biblioteca Matplotlib de Python para graficación.
El código ejemplo consistía en un modelo de comportamiento de población de zorros y conejos, basado en una ecuación diferencial. Como a nosotros nos interesaba un modelo de péndulo simple, se cambió la ecuación, se definieron sus respectivas variables, así como condiciones iniciales y se utilizó el paquete de Matplotlib para generar el retrato fase.

2.1. Resultados

Aquí se presenta el código ya modificado:

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

#Constantillas
```

```
g = 9.81
l = 1.0
b = 0.0
c = g/l


#Codiciones Iniciales
X_f1 =np.array([-3*np.pi,3*np.pi])
X_f2 =np.array([-2*np.pi,np.pi*0])
t = np.linspace(0,20,300)


#Ecuacioncilla

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

#Formatillo
values  =np.linspace(-1.0,1.0,50)
           # position of X0
           between X_f0 and X_f1
vcolors = plt.cm.prism
(np.linspace(0.9, 0.0, len(values)))
# colors for each trajectory
#vcolors8 = plt.cm.cool_r
(np.linspace(0.5, 1.0, len(values)))
 # colors for each trajectory

plt.figure(2)


for v, col in zip(values, vcolors):
y0 = v * X_f1

X = odeint(p, y0, t, args=(b,c))
plt.plot( X[:,0], X[:,1], lw=3.5*v,
 color=col, label='X0=(%.f, %.f)'
  % ( y0[0], y0[1]) )


for v, col in zip(values, vcolors):
y1 = v * X_f2
X1 = odeint(p, y1, t, args=(b,c))
plt.plot( X1[:,0], X1[:,1], lw=3.5*v,
 color=col, label='X0=(%.f, %.f)'
  % ( y1[0], y1[1]) )


#Estúpida gráfica hermosa
plt.title('Trayectorias')
plt.xlabel('Angulo')
plt.ylabel('Velocidad Angular')
plt.grid()
plt.xlim(-2.0*np.pi,2.0*np.pi)
plt.ylim(-4*np.pi,4*np.pi)
```

Al correr el código, obtuvimos la siguiente gráfica:

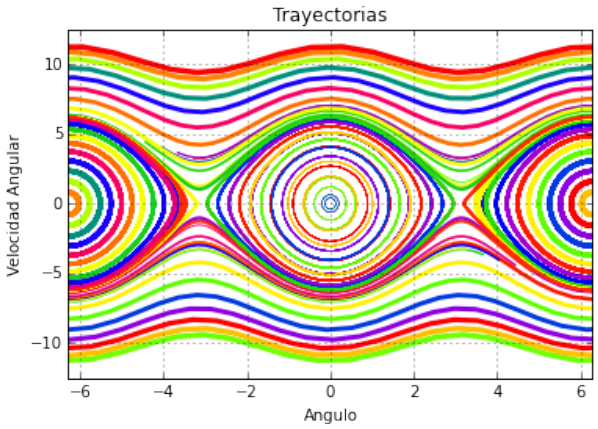

Figura 1: Retrato fase de familia de soluciones del péndulo simple

3. Conclusión

Es importante recalcar que estos retratos son útiles para representar fenómenos en la física. Honestamente, nunca habían sido mencionados en clase por nuestros profesores y es la primera vez que tenemos conocimiento de ellos. Me pareció interesante leer acerca de estas figuras y sus aplicaciones, de tal manera que ahora adquirimos esta nueva herramienta para la representación de fenómenos y todas sus posibles soluciones.