Actividad 7

Andrés Ignacio Rodríguez Mendoza

Introducción

El espacio fase de un sistema dinámico es un espacio en el cual todos los posibles estados del sistema son representados, con cada estado correspondiendo a un punto único en el espacio fase. Para sistemas mecánicos, como el péndulo, el espacio fase consiste en todos los valores posibles de las variables de momento y posición.

Las trayectorias en el espacio fase representa el conjunto de estados compatible con cualquier condición inicial. El diagrama del espacio fase representa todo lo que el sistema puede ser, y su forma puede facilmente elucidar cualidades del sistema que no son triviales.

En mecánica clásica, cada coordenada generalizada para la posición define a un momendo generalizado, que en conjunto definen las coordenadas del espacio fase. En este proyecto se resuelve nuevamente la ecuación del péndulo mediante la herramienta herramienta de la librería Scipy, *integrate.odeint*. Ahora se grafican del vector resultante de la integración las variables que corresponden a la velocidad angular ω y la posición angular θ, siendo θ la coordenada generalizada del sistema y ω su conjugado, para simular el espacio fase del péndulo.

Código

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


def pend(y, t):
    theta, omega = y
    return [omega, -(9.81)*np.sin(theta)]


# === Population equilibrium ===
#
# Before using !SciPy to integrate this system, we will have a closer look on
# position equilibrium. Equilibrium occurs when the growth rate is equal to 0.
# This gives two fixed points:

f2 = p.figure()


t= np.linspace(0, 7 * np.pi, 500)

X_f0 = np.array([-4*np.pi,2*np.pi])
X_f1 = np.array([1*np.pi,-0*np.pi])
```

```
values1 = np.linspace(-5.2, 2, 800)
vcolors1= p.cm.Accent(np.linspace(-1, 1, len(values1)))

values2  = np.linspace(-8, 4, 200)
vcolors2 = plt.cm.nipy_spectral(np.linspace(-0.51 , 0.93, len(values2)))


p.figure(2)


# plot trajectories
for v, col in zip(values1,vcolors1):
    X0 = v * X_f0                               # starting point
    X = integrate.odeint( pend, X0, t)          # we don't need infodict here
    plt.plot( X[:,0], X[:,1], lw=0.5*v, color=col)


#plot trajectories
for v, col in zip(values2, vcolors2):
    X1 = v * X_f1                               # starting point
    Y = integrate.odeint( pend, X1, t)          # we don't need infodict here
    plt.plot( Y[:,0], Y[:,1], lw=0.2*v, color=col)


plt.grid()
plt.xlim(-4*np.pi,4*np.pi)

plt.ylim(-10,10)


# define a grid and compute direction at each point



nb_points   = 20

x = np.linspace(-4*np.pi, 4*np.pi, nb_points)
y = np.linspace(-3*np.pi, 3*np.pi, nb_points)

X1 , Y1  = np.meshgrid(x, y)                        # create a grid


DX= pend([X1, Y1],t)                        # compute growth rate on the gridt

lw= 0.1*y/x
```

```
Q = plt.quiver(X1,Y1,DX[0],DX[1],DX, pivot='mid',cmap=plt.cm.cool,linewidth=lw)

p.title('Trayectorias y direcciones del campo')
p.xlabel('$\Theta$')
p.ylabel('$\omega$')
p.legend()
p.grid()


p.savefig('fase.png', dpi=200)




plt.show()
```

Gáficas