Deber Seminario

Fausto Fabian Crespo Fernandez

Trayectoria del péndulo

El péndulo que debíamos analizar era un péndulo de longitud $l = g$, con fuerza de amortiguación

$-c\dot{\theta}$(Proporcional a la velocidad) y que corresponde a la fricción en el pivote y una fuerza externa periódica $\rho sent(t)$. Entonces como se menciona en la página 54 del libro "Chaos, an introduction to dynamical systems"Alligod.Yorke and Suaer .Srpinger 2000 la ecuación del movimiento es

$$\ddot{\theta} + c\dot{\theta} + sen(\theta) = \rho sen(t)$$

Debido a la fuerza periódica si $\theta(t)$ es solución de esta ecuación diferencial también lo es $\theta(t + 2\pi N) con\ N\ entero.$

Entonces el mapa para tiempo -2π esta bien definido. SI empezamos con $(\theta_0, \dot{\theta}_0)$ en tiempo $t = 0$ y obtenemos $(\theta_1, \dot{\theta}_1)$ luego de tiempo igual a 2π, entonces $(\theta_1, \dot{\theta}_1)$también será el resultado si empezamos con las condiciones iniciales $(\theta_0, \dot{\theta}_0)$ en tiempo $t = 2\pi(o\ 4\pi, 6\pi, 8\pi, ..)$ y siguiendo la ecuación diferencial por tiempo 2π. (Página 56 del libro "Chaos, an introduction to dynamical systems "Alligod.Yorke and Suaer .Srpinger 2000).

Se puede definir un mapa $F: (\theta_i, \dot{\theta}_i) \rightarrow (\theta_{i+1}, \dot{\theta}_{i+1}) con\ i = 0,1,2, \ldots$. Y donde $(\theta_{i+1}, \dot{\theta}_{i+1})$ es la solución de la ecuación diferencial evaluada luego del tiempo 2π a partir de los valores iniciales $(\theta_i, \dot{\theta}_i)$.

F no tienen una forma simple (Página 56 del libro "Chaos, an introduction to dynamical systems "Alligod.Yorke and Suaer .Srpinger 2000) y por eso se eligió el método numérico Runge-Kutta de cuarto orden para resolver la ecuación diferencial en 0 a 2π. Este mencionar que en el artículo http://www.math.cornell.edu/~hubbard/pendulum.pdf se menciona que esta ecuación diferencial es muy sensible al método de integración, selección de paso etc.

Para el método de Runge Kutta definimos las variables $u = \theta, v = \dot{\theta}$ y vector $w = (u, v)$ y

$$w' = (u', v') = (v, -c * v - sen(u) - \rho sen(t))$$

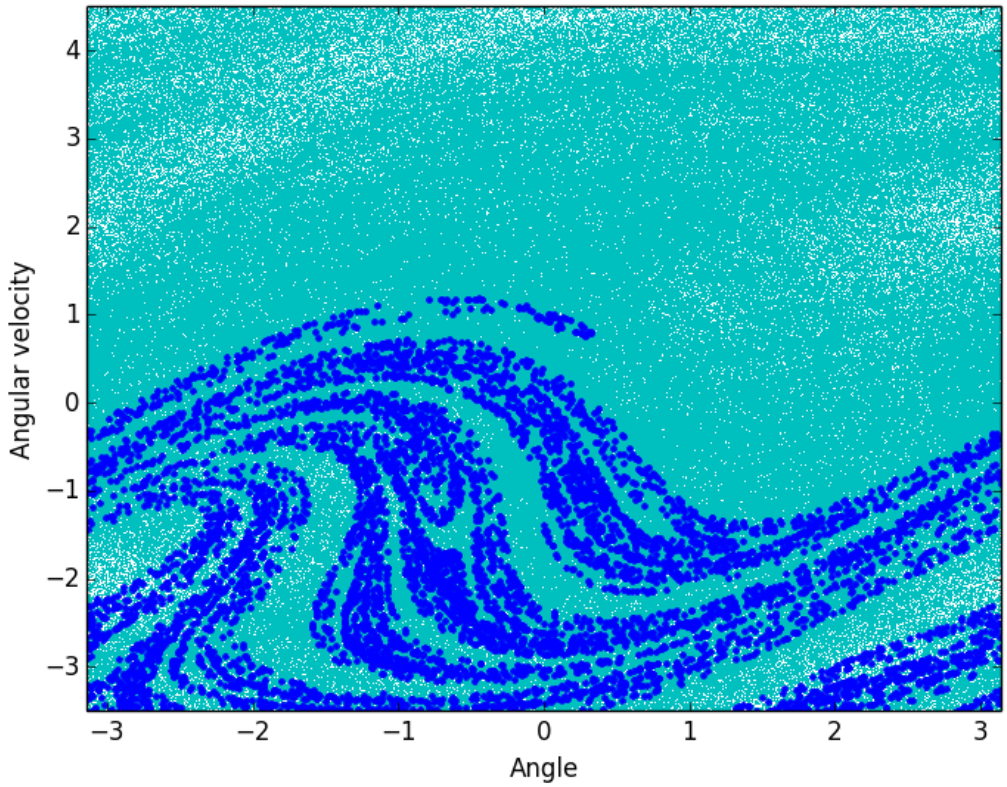
Sin embargo al implementar este método en Python, y hacer la gráfica de velocidad angular vs ángulo del péndulo los resultados no fueron los esperados (un grafica similar a la que aparece en el libro "Chaos, an introduction to dynamical systems "Alligod.Yorke and Suaer .Srpinger 2000 en la página 57)

Al seguir investigando encontré una implementación simple del diagrama de fase (grafica de los parámetros del modelo, en este nuestro caso del péndulo seria velocidad angular vs ángulo del péndulo) del sistema masa resorte en http://compwiki.physics.utoronto.ca/Numerical+Integration .

Existe una explicación muy completa del diagrama de fase para el péndulo amortiguado con fuerza externa periódica en el libro "Computational Physics with python" Dr Eric Ayars(California State University).2013 en http://phys.csuchico.edu/ayars/312/Handouts/comp-phys-python.pdf .

En este libro a partir de la página 109 en el capítulo Chaos se analiza el péndulo simple, el péndulo amortiguado, y el péndulo amortiguado con fuerza externa periódica. Además en el libro se da una implementación en Python para generar el diagrama de fase, en este caso no se usa un método numérico como el método de Euler o el método de Runge-Kutta para resolver la ecuación diferencial sino la función odeint de la librería scipy.integrate de Python

La figura que se obtienen usando el código propuesto en el libro antes mencionado es:



Esta grafica es similar a la del libro de “Chaos, an introduction to dynamical systems”, aunque la gráfica se generó tomando 10000 iteraciones correspondiente a igual número de periodos de tiempo 2π, mientras que la del libro menciona que tiene medio millón de puntos (cada punto representa una iteración del mapa)

Bibliografía

http://www.civilized.com/pdffiles/pendulum.pdf

http://compwiki.physics.utoronto.ca/Numerical+Integration

http://phys.csuchico.edu/ayars/312/Handouts/comp-phys-python.pdf

http://physics.bc.edu/MSC/430/T2/Verlet_DDNLO.html

Código en Python para generar la imagen:

```
from scipy import *
```

```
from scipy.integrate import odeint
from pylab import *
import math

c=0.05
rho=2.5
beta=0.5
g=9.8
length=g
omega =0.9
omega =0.667
A=1.5
def funcPendulum(u, t):
angle, w = u
return (w, -c*w-math.sin(angle)-rho*math.sin(omega_d *t))
def funcPendulum2(u, t):
angle, w = u
return (w, -(g/length)*math.sin(angle)-beta*w + A* cos(omega *t))
def funcPendulum3(u, t):
angle, w = u
return (w, -(g/length)*math.sin(angle)-beta*w)


u0 = array([1,0])
u0 = array([0.5,0])
u0 = array([0,0])
N=1000
t = linspace ( 0 , 25 , N)


u=odeint(funcPendulum2,u0,t)

for i , position in enumerate ( u [ : , 0 ] ) :
while position > math.pi :
position = position - (2.0* math.pi)

while position < (-1 * math.pi) :
position = position + (2.0* math.pi)
u[i,0]=position
figure(1)
plot(t,u[:,0])
xlabel('Time')
ylabel('Angle')
title('Driven forced pendulum')
figure(2)
plot(t,u[:,1])
xlabel('Time')
```

```
ylabel('Angular velocity')
title('Driven forced pendulum')
figure(3)
plot(u[:,0],u[:,1],'-o')
title('Phase-space')
xlabel('Angle')
ylabel('Angular Velocity')
show()

#Poincare plot

steps=100
initial_angle = math.pi / 6
initial_angle = 0
initial_w =0.0
omega_d =1
time_step = 2.0 * math.pi /(omega_d*steps)
N=5000
skip=100
t= arange(0, N * (2* math.pi) / omega_d, time_step)
answer = odeint ( funcPendulum , [initial_angle, initial_w] , t )
answer = answer [ skip * steps : ]

for i , position in enumerate ( answer [ : , 0 ] ) :
while position > pi :
position = position - (2.0* math.pi)

while position < (-1 * math.pi) :
position = position + (2.0* math.pi)
answer[i,0]=position

offset = 50
max_index = (N-skip) * steps - offset

P_thetas = [ ]
P_omegas = [ ]
for j in range ( offset , max_index , steps ) :

P_thetas.append( answer[j, 0] )
P_omegas.append( answer[j, 1] )

figure(4)
plot(answer[:,0], answer[:,1], 'c,')
plot(P_thetas, P_omegas, 'b.')

xlabel('Angle')
ylabel('Angular velocity')
```

```
xlim((-1*math.pi), math.pi)
ylim(-3.5, 4.5)
show( )
```