Universidad de Guadalajara

centro universitario de ciencias exactas e ingenierías

Tarea 3: SDYC

Luis Ramón Mendoza Escareño

Licenciatura en Física | Sistemas Dinámicos y Caos

07 de Marzo del 2023

I. Resumen

Para esta práctica de programación se generó un código para generar un mapa de Poincaré para el caso de dos partículas (q_1,q_2) en un movimiento oscilador armónico simple junto con un factor de acoplamiento entre ellas. Para resolver el problema, se dividió en 3 partes:

II. Función primer punto en el mapeo

Se busca generar una función que entregándole los parámetros de la posición de q_1 y p_1 entregue el punto en la sección de Poincaré (q_2=0, p_2>0), lo cual se consiguió solo despejando p_2 del Hamiltoniano del sistema como sigue:

```
def x0(q_1,p_1,E):
  p_2=np.sqrt(2*m*E-m**2*w**2*q_1**2-p_1**2)
  return (q_1,0,p_1,p_2)
```

Aquí cabe decir, que para encontrar un valor de E que fuera convincente generamos otra función que entregándole valores iniciales de p's y q's nos entregara la energía (esto es posible ya que en este caso la energía la tomamos como constante).

```
def E_c(q_1,q_2,p_1,p_2):
 E=(p_1**2+p_2**2)/(2*m)+ m*w**2/2 *...
 *(q_1**2+q_2**2)+ alpha*q_1**2*q_2**2

 return E
```

III. Generar el siguiente punto

I. Funcion dinámica q's y p's

A partir de tener el primer punto, este se puede tomar como las condiciones iniciales para resolver el sistema de ecuaciones (Cosa que hicimos la tarea pasada), re-utilizando esa función la cual de entrada tiene los valores q,p,Δt nos regresa los valores en el tiempo de cómo evoluciona el sistema a través de q y p (aquí decidimos no poner la función de la tarea pasada para hacer menos bulto cx).

II. Filtrado para cuando q2 cambie de signo

Para localizar el siguiente punto debemos encontrar de el array de la evolución en qué puntos $q_2 = 0$ y p>0, ya que no siempre se acerca q_2 a 0, se usaron las siguientes dos funciones:

```
def Poin(x):
 for i in range(1, len(x)):
  if (x[i-1,1] >= 0 and x[i,1] < 0 and x[i,3] > 0)...
  ...or (x[i-1,1] < 0 and x[i,1] >= 0 and x[i,3]):
      return i
```

Nos entrega la posición donde cambia q_2 de ser positivo a negativo o viceversa y además tiene que cumplir que $p_2 > 0$.

III. Interpolación lineal

Ya que los puntos **no entregan una posición exacta**, se genera una **interpolación lineal** utilizando los puntos q_2 y el tiempo antes y después de cruzar $q_2 = 0$.

```
def cero(q1,q2,t1,t2):
    return (t2-t1)/(q1-q2) * q1 + t1
```

En esta se hace un una función $t(q) = mq + b$, regresa 'b' que es el valor de t(q=0).

IV. Agrupación de las funciones

Finalmente para este paso, se hicieron ciertos arreglos para que se le entregara un vector con $[q_1, p_1, t]$ y entregara el siguiente punto, la función es la siguiente:

```
def Poin_care(vec):
  q_i,p_i,t_a= vec[0],vec[1],vec[2]
  t_f=t_a+10
  q_1,q_2,p_1,p_2 = x0(q_i,p_i,E_1)

  dt=(t_f-t_a)/cut

  q_a=np.append(q_1,q_2)
  v_a=np.append(p_1/m,p_2/m)

  x_0=giro_2(q_a,v_a,t_a,t_f,cut,alpha)
  #Función de la tarea pasada
```

Parte únicamente usada para arreglar la entrada de los valores y que se pueda mandar a la función de la tarea pasada.

```
  a=Poin(x_0)

  q2_1,q2_2=x_0[a,1],x_0[a-1,1]

  t_1, t_2= dt*a , dt*(a-1)   #Para sacar t
                              cuando q2=0

  q1_1,q1_2=x_0[a,0],x_0[a-1,0]  #Valores de
                                q1 y v1  en
 v1_1,v1_2=x_0[a,2],x_0[a-1,2] esos puntos
```

Con estos valores y las interpolaciones lineales encontraremos los puntos para cuando q_2=0

```
  t_00=cero(q2_1,q2_2,t_1,t_2) #Interpolación
                              lineal de t
                    usando los valores de q2
  t_0=t_a+t_00       #El tiempo en que
                         se encontró
  mq=(q1_2-q1_1)/(t_2-t_1)
  mv=(v1_2-v1_1)/(t_2-t_1)
  q_0=mq*t_00+ cero(t_1,t_2,q1_1,q1_2) #Valores q y v
  v_0=mv*t_00+ cero(t_1,t_2,v1_1,v1_2)     #en t00

  return [q_0,v_0*m ,t_0]
```

Se realizan las interpolaciones lineales y se regresa el nuevo punto en la superficie de Poincaré q1,p1 y el tiempo en el cual lo pasa.

IV. Generar los puntos en la superficie

En esta parte únicamente entregamos los valores de las constantes como α, m, ω,...,etc, para poder generar el valor de la energía:

```
alpha=0.5
w=np.pi/2
m=5

cut=80
t0=0
t1=2

q_test=[40,30]
v_test=[8,-10]
E_1=E_c(q_test[0],q_test[1],v_test[0]*m,v_test[1]*m)
```

Ya finalmente generamos una ciclo for con un primer punto en la superficie de Poincaré usando cada salida como entrada de la siguiente iteración, guardamos los puntos en una array para graficarlos.

```
Poin_points=[[25,420,0]]

for i in range(400):
  Poin_points.append(Poin_care(Poin_points[i]))

Poin_points=np.array(Poin_points)
```

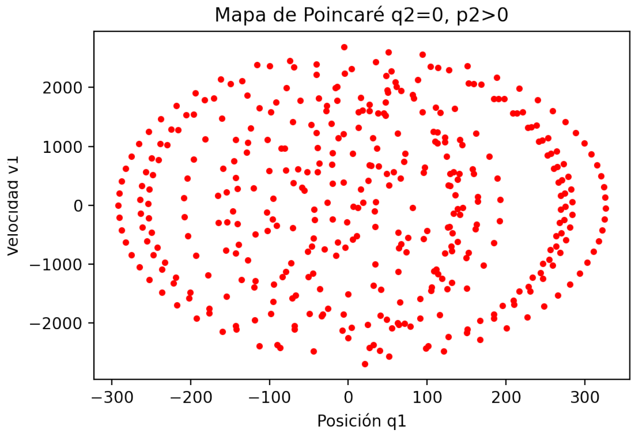

Figura 1: Mapa de Poincaré con 400 elementos dentro de él

De igual manera, queda el acceso al código en google colab.