Universidad de Guadalajara

centro universitario de ciencias exactas e ingenierías

Tarea 4: SDYC

Luis Ramón Mendoza Escareño

Licenciatura en Física | Sistemas Dinámicos y Caos

24 de Marzo del 2023

I. Resumen

En la práctica se calculó analíticamente los puntos de intersección de una trayectoria con movimiento rectilíneo uniforme y ciertas fronteras. Además de adaptar el código del mapeo hacia la superficie de Poincaré y se tomaron dos casos específicos $q_1 = q_2$,$p_1 = p_2$ y $q_1 = 0$,$p_1 = 0$ pertenecientes a trayectorias 'fundamentales' las cuales se calcularon después de ciertas perturbaciones.

Pd: La parte del cálculo analítico lo dejaremos hasta el final para primero generar el documento en látex.

II.

I. a)

Para hacer que se pudieran tomar varios puntos sólo se generó la siguiente función:

```
  def extra_points(vec_2):
a=[]
for i in range(len(vec_2)):
  a.append(Poin_care(vec_2[i]))
return a
```

En la cual sólo se tienen que poner los puntos en el mapa de poincaré de la siguiente forma: $[[\vec{p_1}, \vec{p_2}, ..., \vec{p_n}]]$ y regresa la misma lista anidando más listas dentro con cada uno de los elementos mapeado al siguiente punto en la superficie de Poincaré.
Nota: Poin_care() es la función que definimos en la tarea pasada, la cual por cierto ya se arregló.

II. b)

Para esta segunda parte, primero se confirma el funcionamiento correcto del mapeo de Poincaré con los puntos (0,0) y $(0,\sqrt{mE})$, de la cual, al aplicar el mapeo 6 veces nos quedó de la siguiente manera:

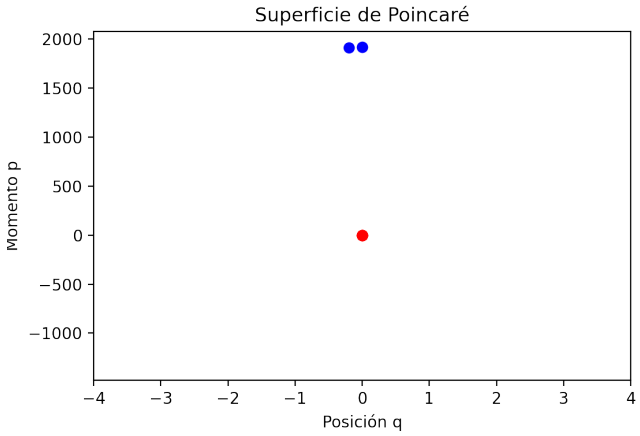

Figura 1: Mapa de Poincaré con los primeras 6 iteraciones para los puntosde las trayectorias fundamentales,(puntos rojos pertenece a (0,0) y los azules a $(0,\sqrt{mE})$).

```
    #Primero nos aseguramos que funcione el mapeo
os_arm=[[0,0,0]]
q_p=[[0,np.sqrt(m*e_c),0]]

for i in range(6):
  os_arm.append(Poin_care(os_arm[i]))
  q_p.append(Poin_care(q_p[i]))

os_arm=np.array(os_arm)
q_p=np.array(q_p)
```

II.1. Linearización

Para esta parte,se calculó la matriz de linearización a partir de:

$$L(x_0 + h\vec{v}) \sim x_0 + hL(\vec{v}) \tag{1}$$

De aquí que:

$$L(\vec{v}) \sim \frac{L(x_0 + h\vec{v}) - x_0}{h} \tag{2}$$

Despejando para $L(\vec{v})$, para esto primero se encuentra un valor para h. Observando que el error entre el valor esperado para el periodo y el generado por el programa fue de $\sim 8 \cdot 10^{-3}$, se propone una h de tamaño 0.01.
Seguidamente se calculó el mapeo con la perturbación en 'q' y 'p' , y a partir de estos se generaron las matrices L para ambos puntos.

```
dp=0.01
dq=0.01
os_q=[0+dq,0,0]
os_p=[0,0+dp,0]

pq_p=[0,np.sqrt(m*e_c)+dp,0]
pq_q=[0+dq,np.sqrt(m*e_c),0]

h=0.01

#Para p=q
l1_pq=np.array(pq_q)[:2] -

np.array(Poin_care(pq_q))[:2]/ h

l2_pq=np.array(pq_p)[:2] -
np.array(Poin_care(pq_p))[:2] / h

#Para el oscilador
l1_os=np.array(os_q)[:2]-
np.array(Poin_care(os_q))[:2] / h

l2_os=np.array(os_p)[:2] -
np.array(Poin_care(os_p))[:2] / h
```

Finalmente se usan estos vectores para generar las matrices y se calculan los eigenvectores y eigenvalores.

```
L1=np.column_stack((l1_os,l2_os))
L2=np.column_stack((l1_pq,l2_pq))

eigenvalues_1, eigenvectors_1 = np.linalg.eig(L1)
eigenvalues_2, eigenvectors_2 = np.linalg.eig(L2)
```

Para el caso de **oscilador armónico**, los eigenvalores fueron:
$[0.6422 + 0.7715j, 0.6422 - 0.7715j]$
Complejos conjugados, por tanto se tiene una trayectoria estable que oscila alrededor del punto de equilibrio.

Por otra parte para el caso de los valores cuando $p_1 = p_2, q_1 = q_2$, se tiene que los eigenvalores fueron: [-56.6257830, -189897.804] , ambos negativos, lo cual indica que las direcciones de los eigenvectores:

$$\begin{pmatrix} 0.704901724 & .00001548863 \\ -0.709304983 & 1 \end{pmatrix} \tag{3}$$

Son direcciones estables, lo cual no pareciera concordar con las imágenes que generadas en la práctica pasada, ya que se observaba que con pequeñas perturbaciones cambia un poco en q, pero mucho en p (justo como en el segundo eigenvector) y parecían alejarse del punto de equilibrio, no acercarse, esto me parece raro.

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