Tarea 1 SDYC: implementación en python

Luis Ramón Mendoza Escareño

Enero 2023

```
import numpy as np
import matplotlib.pyplot as plt

,,,
 q --> [cm]
p --> [kg*cm/s]
w --> [1/s]
m --> [kg]
t --> [s]

Se quiere resolver
dq/dt = p/m
dp/dt = -mqw**2
,,,
#Asignación de valores iniciales y constantes
q0=5
p0=50
w=4
m=5
#Implementación de los valores reales que toman p0 y q0 extraidos del Hamiltoniano
#Ahora calculamos el valor de la energía total con el Hamiltoniano con las C.I
En=p0**2/(2*m)+ m*w**2*q0**2/2

def H(q,p): return p**2/(2*m) + m*w**2*q**2/2 - En #La función que nos da la
                               verdadera curva en el espacio de fases

ax_1= np.sqrt(2*En*m)   #Los valores de los semiejes de p y q
ax_2= np.sqrt(2*En/(m*w**2))

p_b=np.linspace(-ax_1,ax_1,300)  #El intérvalo de p y q partido en 300 puntos
q_b=np.linspace(-ax_2,ax_2,300)

x,y=np.meshgrid(q_b,p_b) #La unión de estos intérvalos en un array
z=H(x,y)                 #Z evaluada en estos puntos
```

```
def Pos(i):  #Entrega los valores de p y q en función de un argumento que disminuye dt
  steps=10000       #Iteraciones del algoritmo
  M=np.array([[0,1/m], [-w**2 * m,0]]) #Matriz con los coeficientes de p y q
  x_0=np.array([q0,p0])                #Condiciones inciales
  X=np.zeros((2,steps+1))              #Matriz donde se guardarán como irá cambiando X
  X[:,0]=x_0                      #Ponemos la primer columna como los valores iniciales
  for j in range(steps):
    dt=1/(100*(i+1))
    x_0=x_0+dt*np.matmul(M,x_0)          #Algoritmo de Euler
    X[:,j+1]=x_0
  return X

#Ploteo de la función (1)
fig, ax= plt.subplots(2,2)
for i in range(2):
  for j in range(2):

    X_2=Pos(i*4+j*8+3)
    dt=1/(100*(i*4+j*8+3))

    ax[i,j].contour(x,y,z,[0])
    ax[i,j].plot(X_2[0,:],X_2[1,:],color='b')
    fig.tight_layout(pad=2.0)
    ax[i,j].set_title('dt = '+str(round(dt,5)))


    #En esta parte se evalúa la función para varios dt y los ponemos
#Todos juntos
X_1=Pos(3)
plt.plot(X_1[0,:],X_1[1,:],color='b',label='dt=0.0033')
X_2=Pos(7)
plt.plot(X_2[0,:],X_2[1,:],color='g',label='dt=0.0014')
X_3=Pos(11)
plt.plot(X_3[0,:],X_3[1,:],color='r',label='dt=0.0009')
X_4=Pos(15)
plt.plot(X_4[0,:],X_4[1,:],color='y',label='dt=0.0006')


plt.xlabel('Posición [cm]' )
plt.ylabel('Momentum [kg*cm/s]')
plt.legend()
plt.show()
```

En las siguientes figuras se tomaron 10,000 pasos en el algoritmo iterativo de Euler, únicamente cambiando los valores del factor de cambio 'dt', se realizaron de 2 maneras diferentes:

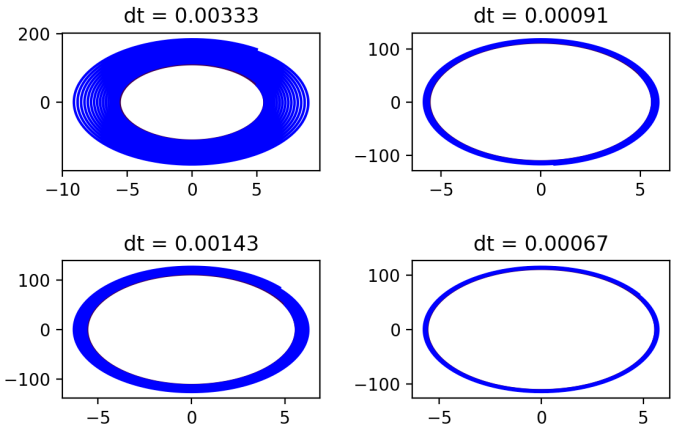

Figure 1: En este caso, para cada valor diferente de dt se realizó una gráfica, se observa que a medida que el dt disminuye, las gráficas parecieran más cercanas a una elipse.

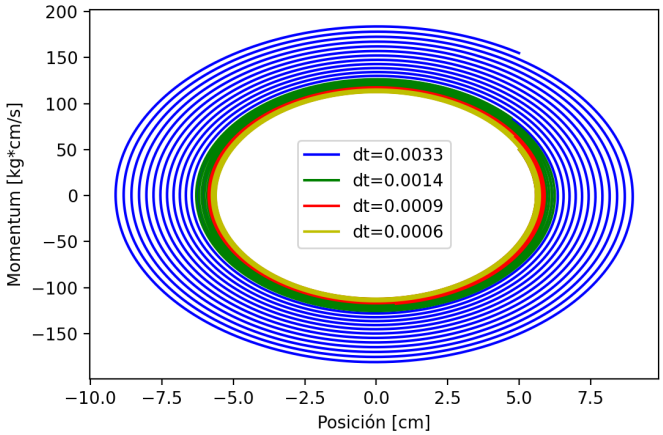

Figure 2: Para esta figura, se tomaron los valores de dt antes vistos, con la diferencia que aquí se pusieron en una sola gráfica, permitiendo ver de mejor manera la diferencia entre ellos.

De igual manera dejo aquí el link en google colab