Actividad #6 (Periodo del Péndulo)

Instructor: Carlos Lizárraga Celaya *Student: Antonio Cota Rodríguez*

Introducción

Como ya vimos en nuestra primera actividad, la matemática del péndulo es un poco complicada ya que nos queda por resolver una ecuación diferencial de segundo orden no lineal homógenea :

$$\frac{d^2\theta}{dt^2} + \frac{g}{l}\sin\theta = 0$$

y ya se ha visto que podemos resolver la ecuación anterior haciendo que $\sin\theta \approx \theta$ esto significa que consideraremos solo oscilaciones pequeñas, esto nos lleva a que la solución queden en términos de funciones elementales :

$$\theta(t) = A\cos(\omega t + \delta) \qquad A \ll 1$$

donde esta es la ya conocida ecuación del oscilador armónico, con periodo de oscilación :

$$T_0 = 2\pi\sqrt{\frac{l}{g}}$$

Pero lo que nos concierne ahora es estudiar la solución de la ecuación diferencial para **amplitudes arbitrarias** (veáse actividad 1 "Deducción de la ecuación por el método de la energía"), tenemos la siguiente ecuación diferencial :

$$\frac{d\theta}{dt} = \sqrt{\frac{2g(\cos\theta - \cos\theta_0)}{l}}$$

Se invierte la ecuación y se integra sobre un ciclo completo, así nos queda resolver la siguiente integral :

$$T = 4\sqrt{\frac{l}{2g}}\int_0^{\theta_0}\frac{d\theta}{\sqrt{(\cos\theta - \cos\theta_0)}}$$

Obteniéndo el cociente de los dos periodos T/T_0 nos da la desviación del periodo verdadero del péndulo para una aproximación de oscilaciones pequeñas.

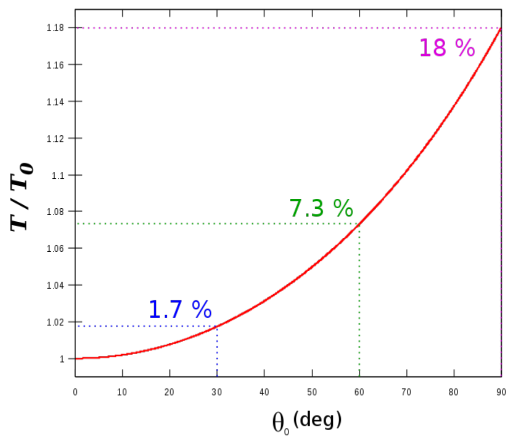

Figure 1 – Desviación para pequeñas oscilaciones

notar que $\lim_{\theta_0\to\pi} T = \infty$.

Programa

En esta actividad se realizó un código en ipython con objetivo de resolver la integral para el periodo del péndulo con un error mínimo así relacionando en una gráfica el error relativo y θ_0.

```
# Paquetes
import numpy as np
from scipy import integrate
import matplotlib.pyplot as plt


# Numero de angulos a integrar
n=1000

# Arreglos, tomando error de 0.0001
x=[]
TT0=[]
x_0=np.linspace(0.0001,np.pi + 0.0001, n)



# La funcion a integrar
I = lambda x,a: 1/np.sqrt(np.cos(x) - np.cos(a))


for i in range(n):
# Integral
    theta_0=x_0[i]
    T , err= integrate.quad(I, 0, theta_0, args=(theta_0,))


# Error
    TT0.append(np.sqrt(2)/np.pi * T)



# Graficas
plt.figure(1)
plt.plot(x_0 * 180 / np.pi, TT0 , "r" )
plt.title('Desviacion')
plt.xlabel(r'$ \theta _0 (grados)$')
plt.ylabel("T/To")
plt.xlim(0,90)
plt.ylim(1,1.2)
plt.grid()

plt.show()

plt.figure(1)
plt.plot(x_0 * 180 / np.pi, TT0 , "b" )
plt.title('Divergencia en ' r'$\theta_0 = \pi$')
plt.xlabel(r'$ \theta _0 (grados)$')
plt.ylabel("T/To")
plt.xlim(0,180)
plt.ylim(1,5)
plt.grid()


plt.show()
```

Para $n = 1000$ y $\epsilon = 0.0001$ obtuvimos las siguientes gráficas :

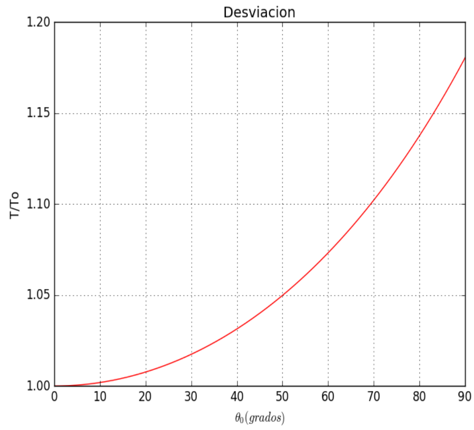

FIGURE 2 –

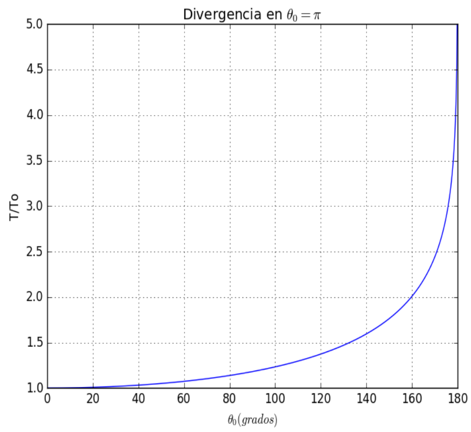

FIGURE 3 –

Conclusión

Pudimos comprobar gráficamente como diverge T/T_0 cuando $\theta \to \pi$ utilizando el método para integrar de **scipy.integrate.qua**