Actividad 6: Período del péndulo

Ana Gabriela Carretas Talamante

01 de marzo de 2016

1. Introducción

Anteriormente hablamos sobre la ecuación diferencial que describe el movimiento del péndulo y la resolvimos utilizando una librería especial de Python con condiciones de ángulos pequeños. En esta ocasión trabajaremos con la integral que muestra sobre el período del péndulo, haciendo variar el ángulo inicial sin esa restricción.

Podemos calcular el período exacto invirtiendo la ecuación de velocidad angular:

$$\frac{d\theta}{dt} = \sqrt{\frac{2g}{l}(\cos\theta - \cos\theta_0)} \tag{1}$$

$$\frac{dt}{d\theta} = \sqrt{\frac{l}{2g}}\frac{1}{\sqrt{(\cos\theta - \cos\theta_0)}} \tag{2}$$

Y luego integrando 4 veces un cuarto de ciclo, lo que nos dejaría la integral a resolver:

$$T = 4\sqrt{\frac{l}{2g}}\int_0^{\theta_0}\frac{1}{\sqrt{(\cos\theta - \cos\theta_0)}} \tag{3}$$

Encontramos el error relativo al dividir el resultado de (3) entre el período inicial:

$$T_0 = 2\pi\sqrt{\frac{l}{g}} \tag{4}$$

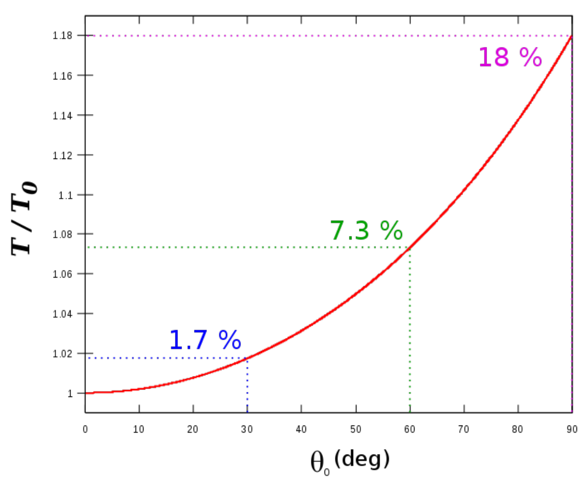

Figura 1: Desviación entre el período real y el calculado con la aproximación de ángulos pequeños mientras θ_0 aumenta [1].

Periodo de oscilación «exacto» del péndulo simple para diversas amplitudes de oscilación.

θ	T/T_0	θ	T/T_0	θ	T/T_0	θ	T/T_0
0°	1.0000	20°	1.0076	40°	1.0313	100°	1.2322
2°	1.0001	22°	1.0093	44°	1.0381	110°	1.2953
4°	1.0003	24°	1.0193	48°	1.0457	120°	1.3729
6°	1.0007	26°	1.0131	52°	1.0541	130°	1.4693
8°	1.0012	28°	1.0152	56°	1.0632	140°	1.5945
10°	1.0019	30°	1.0174	60°	1.0132	150°	1.7737
12°	1.0027	32°	1.0199	70°	1.1021	160°	2.0075
14°	1.0038	34°	1.0225	80°	1.1375	170°	2.4393
16°	1.0049	36°	1.0253	90°	1.1804	180°	∞
18°	1.0062	38°	1.0282				

Figura 2: Valores de desviación entre el período real y el calculado con la aproximación de ángulos pequeños mientras θ_0 aumenta [2].

Mientras el valor de θ_0 crezca hasta acercarse a valores donde $\theta_0 \sim \pi$, los resultados del valor relativo del período divergen a ∞, como se muestra en las figuras 1 y 2.

El método *scipy.integrate.quad* [3] nos ayuda a resolver integrales con la librería QUADPACK de Fortran. Este trabaja analizando la pendiente de la función que va a integrar, relacionándola con el intervalo que va a utilizar para la integración numérica para maximizar la eficiencia del cálculo [4].

Programa: Resolviendo la integral para el período del péndulo Se presenta a continuación el código realizado, donde variamos el ángulo inicial desde un error e hasta el ángulo $\pi + e$. Esto se hizo para evitar que tuviera una integral con 0 en el cociente [5]. Utilizando el método para integrar de *scipy.integrate.quad*, se pudo calcular el valor del período para cada valor de θ_0 indicado por el array ya definido. Al finalizar estos cálculos, se obtuvo el error relativo entre el período calculado y el inicial, y con esto se graficó a $\theta_0 \quad \text{vs} \quad \dfrac{T}{T_0}$.

```
#Bibliotecas utilizadas
from scipy.integrate import quad
import numpy as np
import matplotlib.pyplot as plt

#-------------------------------------------
#Definimos las constantes
#Valor de la gravedad
g=9.8
#Longitud de la cuerda
l=0.5
#Periodo inicial
T0=2*np.pi*np.sqrt(l/g)
```

```
#Constante de la integral para angulos pequeños
k=4*np.sqrt(l/(2*g))

#------------------------------------------
#Defino los posibles angulos para integrar
n=100
#Error añadido para que no divida entre 0
e=0.0001
#Rango de theta0
theta00=np.linspace(e,(np.pi)+e,n)

#------------------------------------------
#Defino los arrays para todos los resultados arrojados
II=[0 for i in range(n)]
err=[0 for i in range(n)]
T=[0 for i in range(n)]

#------------------------------------------
#Definiendo la integral
def I(x, theta0):
    return 1/np.sqrt(np.cos(x)-np.cos(theta0))

#Comienzo un loop para poder calcular todos los resultados
#posibles contemplando un angulo inicial variante
for i in range(n):
    theta0=theta00[i]
    II[i] , err[i]=quad(I, 0, theta0, args=(theta0))
    T[i]=k*II[i]

#Calulo el periodo real entre el
#periodo calculado y el inicial
real=T/T0

#------------------------------------------
#Para la grafica
plt.plot(theta00, real, "mo", theta00, real, "g")
plt.title("Desviacion del periodo real con respecto al angulo")
plt.grid()
plt.xlabel("Angulo en radianes")
plt.ylabel("Periodo real")
```

Analizando las gráficas

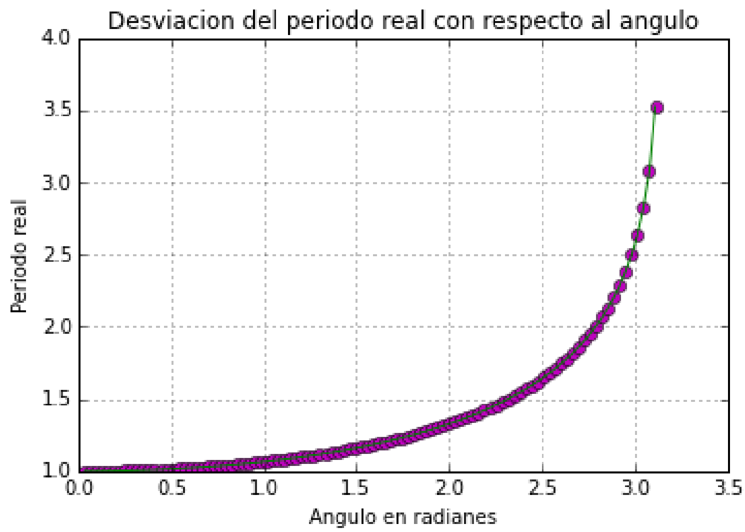

Figura 3: Gráfica generada para n=100.

Podemos notar que mientras aumento el tamaño de las particiones del intervalo $(0, \pi)$, el error relativo tiende a ∞, como se había mencionado antes.

Figura 4: Gráfica generada para n=1000.

Figura 5: Gráfica generada para n=1500.

Figura 6: Gráfica generada para n=2000.

Referencias

[1] Alessio Damato, *Pendulum period.* Recuperado en marzo de 2016 de https://en.wikipedia.org/wiki/Pendulum_%28mathematics%29#/media/File:Pendulum_period.svg

[2] Algarabia, *Pendulo Simple - Tabla exacta.* Recuperado en marzo de 2016 de https://commons.wikimedia.org/wiki/File:Moglfm13t02_pendolo_simple_exacto.jpg

[3] Scipy, *quad.* Recuperado en febrero de 2016 de http://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.quad.html#scipy.integrate.quad

[4] Stackoverflow, Recuperado en marzo de 2016 de http://goo.gl/OOKoYY

[5] Paredes M., *Asesoría sobre el código*, consultado en febrero de 2016.

[6] Lizárraga, C. *Actividad 6 (2016-1).* Recuperado en febrero de 2016 de http://computacional1.pbworks.com/w/page/105233358/Actividad%205%20(2016-1)