Q.1.1

Q.1.1.1

$T = \frac{1}{2}mv^2 = \frac{1}{2}mR^2\dot{\theta}^2$

$V = -mgRcos\theta$

Where, $v = R\theta$

$L = T - V$ Lagrangian formula

$$L = \frac{1}{2}mR^2\dot{\theta}^2 + mgRcos\theta$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \left(\frac{\partial L}{\partial \theta}\right) = 0$$ Lagrange Formula

$$mR^2\ddot{\theta} + mgRsin\theta = 0$$

$$\ddot{\theta} + \frac{g}{R}sin\theta = 0$$ For small osscilation $\sin\theta = \theta$

$$\ddot{\theta} + \frac{g}{R}\theta = 0$$

Thus the frequency is $\omega = \sqrt{\frac{g}{R}}$

Q.1.1.2

5:49 PM

$T_{translational} = \frac{1}{2}mR^2\dot{\theta}^2$

$T_{rotational} = \frac{1}{2}I\dot{\theta}^2 = \frac{1}{5}mR^2\dot{\theta}^2$

$V = -mgRcos\theta$

$I_{ball} = \frac{2}{5}mR^2$

$L = T - V$

$$L = \frac{1}{2}mR^2\dot{\theta}^2 + \frac{1}{5}mR^2\dot{\theta}^2 + mgRcos\theta$$

$$L = \frac{7}{10}mR^2\dot{\theta}^2 + mgRcos\theta$$

Q.1.1.3

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \left(\frac{\partial L}{\partial \theta}\right) = 0$$

$$\frac{14}{10}mR^2\ddot{\theta} + mgRsin\theta = 0$$

$$\ddot{\theta} + \frac{5g}{7R}\theta = 0$$

Thus the frequency is $\omega = \sqrt{\frac{5g}{7R}}$

Q.1.2 RJ

Q.1.2.1 RJ

$M_r = \frac{4}{3}\pi r^3 \rho$

$F = -\frac{GM_r m}{r^2} = -G\frac{4}{3}\pi r \rho m$

$Sub\ in\ desnity\ expression\text{:}\ \rho = \frac{3M}{4\pi R^3}$

$F = -\frac{GMmr}{R^3}$

Q.1.2.2 RJ

F = ma

$m\ddot{r} + \frac{GMmr}{R^3} = 0$

$\omega = \sqrt{\frac{GM}{R^3}}$

From, $T = \frac{2\pi}{\omega}$

$Thus\ the\ period\ is\text{:}\ T = 2\pi\sqrt{\frac{R^3}{GM}}$

Q.1.2.3 RJ

M =5.972x10^4 kg

T = 5.06 x 10^3s = 84.3 min

$v_{peak} = \frac{2\pi A}{T}\ where\ A = R\ where\ R = 6.371x10^6 m$

Thus using period from above: $v_{peak} = \frac{2\pi R}{T} = 7.91x10^3 m/s$

RJ

Q.1.3

Q.1.3.1

Overdamped Solution: (Two Real Roots)

$x(t) = C_1e^{r1t} + C_2e^{r2t}$ General Solution

$0 = C_1e^{r1t} + C_2e^{r2t}$

$C_1e^{r1t} = -C_2e^{r2t}$

$\frac{C_1}{C_2} = -\frac{e^{r2t}}{e^{r1t}}$

$(\mathrm{r2} - \mathrm{r1})\ln\left(-\frac{C_1}{C_2}\right) = \mathrm{t}$

Therefor the crossing occurs when $-\frac{C_1}{C_2}$ is greater than 1 to avoid undefined solution.

Critically Damped Solution: (One Real Root)

$x(t) = (C_1+tC_2)e^{rt}$ General Solution

$0 = (C_1+tC_2)e^{rt}$

$-\frac{C_1}{C_2} = t$

Both these solutions can cross the origin at most once because they following the shape of an exponential. Neither solution included oscillatory trigonometric functions and so no oscillation will take place and thus the solutions can only cross the x-axis at most once.

Q.1.3.2

Initial Conditions: $X(0) = x_0$, $X'(0) = v_0$

Where r can be written as $\frac{\alpha}{2} = -\omega$

$x(t) = (C_1+tC_2)e^{rt}$
$x(0) = C_1$

$\frac{d}{dt}x(t) = \frac{d}{dt}(C_1+tC_2)e^{rt} = C_2e^{rt} + (C_1+tC_2)re^{rt}$
$v_0 = C_2e^{rt} + (x_0 + tC_2)re^{rt}$

$v_0 = C_2 + x_0r$
$v_0 = C_2 - \omega x_0$

For the oscillator to no cross the x-axis the speed $|v_0|$!= ωx_0 , so the range of speeds can be $|v_0| < \omega x_0$

1.4.1

Under-damped harmonic oscillator

$$x(t) = x_0 e^{-\alpha t} \cos(\omega_d t + \phi)$$

$$\omega_d = \sqrt{\omega^2 - \alpha^2}$$

For half-life:

$\phi = 0$

$$\frac{\cancel{x_0}}{2} = \cancel{x_0} e^{-\alpha t} \cos(\omega_d t + \phi)$$

$$\frac{1}{2} = e^{-\alpha t} \cos(\omega_d t)$$

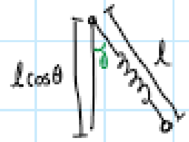
Solve for

$$\frac{1}{2} = e^{-\alpha t}$$

$$\alpha t = 2 \ln(2)$$

$$\boxed{t = \frac{2 \ln(2)}{\alpha}}$$

1.5.1 $\quad L = T - U$



$$V = V_r + V_\theta = \frac{dr}{dt} + r\frac{d\theta}{dt} = \dot{\ell} + \ell\dot{\theta}$$

In Polar Coordinates

$$T = \frac{1}{2}mv^2 = \frac{1}{2}m(\dot{\ell}^2 + \ell^2\dot{\theta}^2)$$

$$U = -mgh + \frac{1}{2}Kx^2 = -mg\ell\cos\theta + \frac{1}{2}K(\ell - \ell_0)^2$$

$$L = T - U$$
$$L = \frac{1}{2}m(\dot{\ell}^2 + \ell^2\dot{\theta}^2) + mg\ell\cos\theta - \frac{1}{2}K(\ell - \ell_0)^2$$
$$L = \frac{1}{2}m\dot{\ell}^2 + \frac{1}{2}m\ell^2\dot{\theta}^2 + mg\ell\cos\theta - \frac{1}{2}K(\ell - \ell_0)^2$$

Using Euler-lagrange Formula

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\ell}} = \frac{\partial L}{\partial \ell} \quad \text{and} \quad \frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}} = \frac{\partial L}{\partial \theta}$$

For ℓ

$$\frac{\partial L}{\partial \dot{\ell}} = m\dot{\ell} \qquad \frac{\partial L}{\partial \ell} = m\ell\dot{\theta}^2 + mg\cos\theta - K(\ell - \ell_0)$$

$$\frac{d}{dt}[m\dot{\ell}] = m\ell\dot{\theta}^2 + mg\cos\theta - K(\ell - \ell_0)$$

$$m\ddot{\ell} = m\ell\dot{\theta}^2 + mg\cos\theta - K(\ell - \ell_0)$$

$$\ddot{\ell} - \ell\dot{\theta}^2 - g\cos\theta - \frac{K}{m}(\ell - \ell_0) = 0$$

For θ

$$\frac{\partial L}{\partial \dot{\theta}} = m\ell^2\dot{\theta} \qquad \frac{\partial L}{\partial \ell} = -mg\ell\sin\theta$$

$$\frac{d}{dt}[m\ell^2\dot{\theta}] = -mg\ell\sin\theta$$

$$2m\ell\dot{\ell}\dot{\theta} + m\ell^2\ddot{\theta} = -mg\ell\sin\theta$$

$$\ddot{\theta}\ell + 2\dot{\ell}\dot{\theta} + g\sin\theta = 0$$

$$\ddot{\theta} + \frac{2}{\ell}\dot{\ell}\dot{\theta} + \frac{g}{\ell}\sin\theta = 0$$

$\therefore$ The two equations are

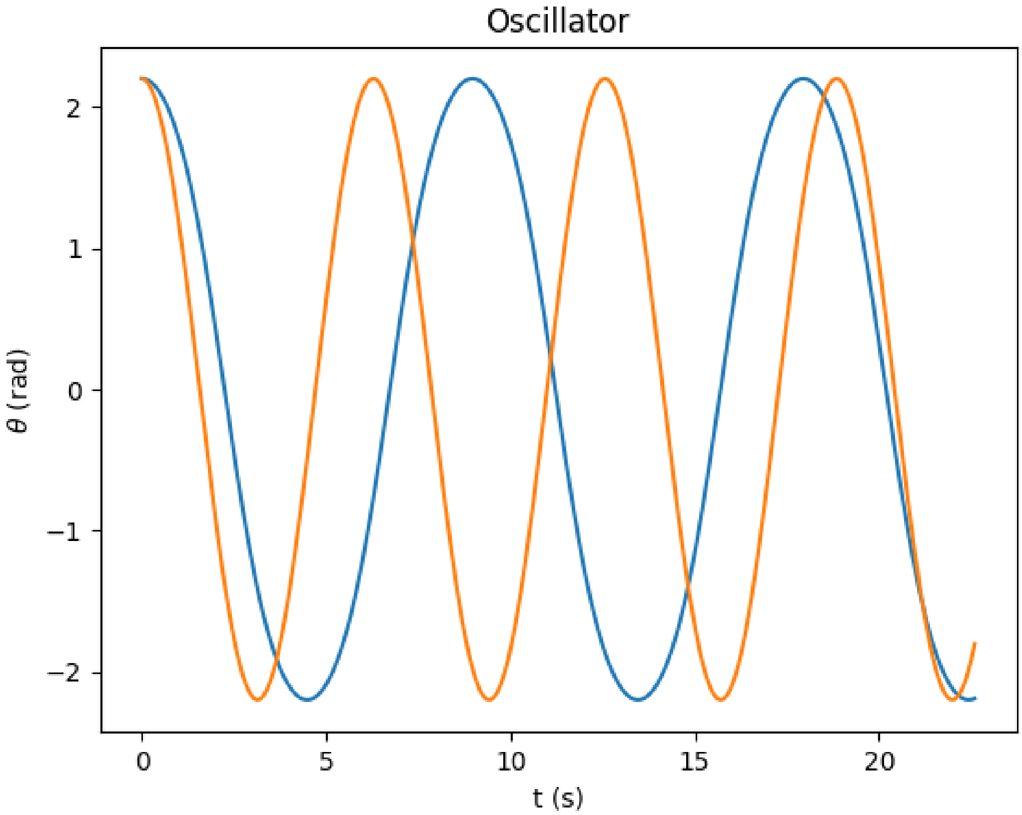
$$\boxed{\begin{aligned} &\ddot{\ell} - \ell\dot{\theta}^2 - g\cos\theta - \frac{K}{m}(\ell - \ell_0) = 0 \\ &\ddot{\theta} - \frac{2}{\ell}\dot{\ell}\dot{\theta} + \frac{g}{\ell}\sin\theta = 0 \end{aligned}}$$

```
#!/usr/bin/env python
# coding: utf-8

# ENPH 211 / PHYS 212 2024W
# ====
# Student names:
# ------
# Submit your code, a pdf of the code, and pdfs of any figures you make.  Ensure
your figures are labelled and titled.
# ------

# The following code solves the second order linear differential
equation  $\ddot{\theta} + \ \omega_0^2 \ \sin \theta = 0$.


#%% Load libraries
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt



#%% Parameters
omega_0_rads = 1.0 # Natural angular frequency

theta0_rad, thetaDot0_rads = 0.7 * np.pi, 0.0 # Initial position and velocity

outpath = "./figure_" # Where figure goes



#%% Do the numerical integration
def dU_dt(U, t):
    # Here U is a vector such that y=U[0] and z=U[1]. This function should return
[y', z']
    return [U[1], - omega_0_rads**2 * np.sin(U[0])]

U0 = [theta0_rad, thetaDot0_rads]
ts, tIncrement_s = np.linspace(0, 9 * np.pi / omega_0_rads, 300, retstep=True)
Us = odeint(dU_dt, U0, ts)
thetas = Us[:,0]



#%% Plot
plt.xlabel("t (s)")
```

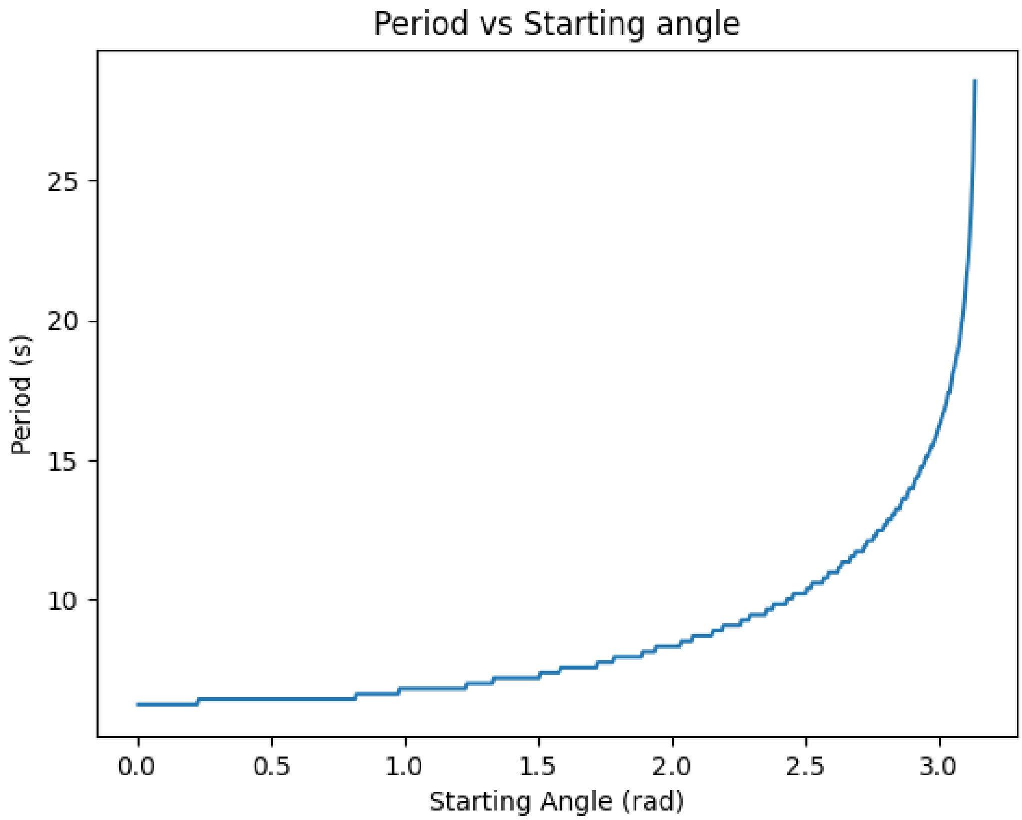
```
plt.ylabel(r"$\theta $ (rad)")
plt.title("Oscillator")
plt.plot(ts, thetas, label='pos')

plt.show(); plt.clf()


#%%

# Q1) (3 pts) Numerically determine the period of the previous oscillator,
# for instance by identifying consecutive samples of opposite sign.
# ----


# Here's the cheater method of using an already nicely created scipy function
from scipy.signal import find_peaks

def scipyPeriod (thetas):
    peaksInd, _ = find_peaks(thetas)
    return ts[peaksInd[1]] - ts[peaksInd[0]]

print(f"Period of the given oscillator using Scipy is {scipyPeriod(thetas):.2f}
seconds")

# Here's the suggested method

def getApproxPeriod(thetas):
    currentState = None
    switchState = None

    switchIndices = []
    for i in range(len(thetas)):
        if thetas[i] < 0: currentState = False
        else: currentState = True # Checking +/- state of theta value

        if currentState != switchState: # we only care about +/- transition
points so check current state of function. True is pos, False is neg
            switchIndices.append(i)
            switchState = currentState

    return (switchIndices[2] - switchIndices[1])*2*tIncrement_s # Taking index 3
and 1 because first switch index will always be zero the way that this has been
coded (because going from none to True or False is a state switch)
```

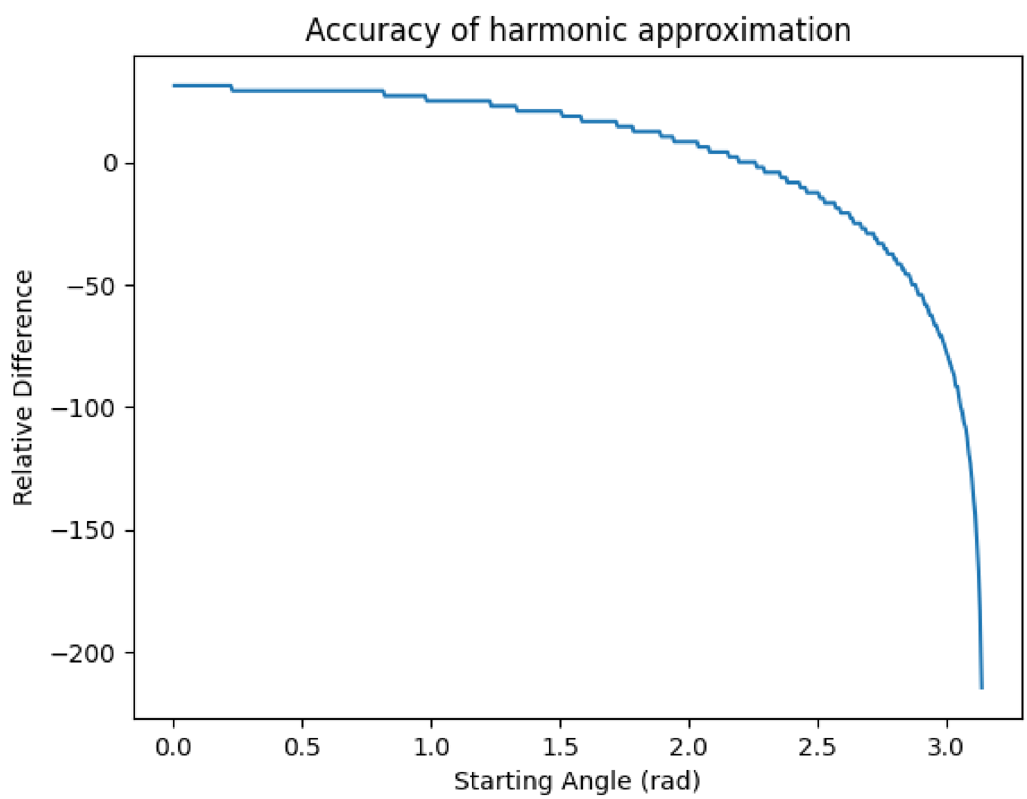
```
print(f"Period of the given oscillator using sign switching is
{getApproxPeriod(thetas):.2f} seconds")


#%%
# Q2 (3 pts) Reproduce the previous plot over roughly 2 periods with the harmonic
approximation overlayed.
# ---


def smallAngleApproxdU_dt(U, t):
    # Here U is a vector such that y=U[0] and z=U[1]. This function should return
[y', z']
    return [U[1], - omega_0_rads**2 * U[0]]

Us = odeint(dU_dt, U0, ts[:int((4/5)*300)])
Uthetas = Us[:, 0]

sAAUs = odeint(smallAngleApproxdU_dt, U0, ts[:int((4/5)*300)])
sAAThetas = sAAUs[:, 0]

plt.xlabel("t (s)")
plt.ylabel(r"$\theta $ (rad)")
plt.title("Oscillator")

plt.plot(ts[:int((4/5)*300)], Uthetas)
plt.plot(ts[:int((4/5)*300)], sAAThetas)


plt.show(); plt.clf()


#%%
# Q3 (3 pts) Determine the relative difference between the period and the
harmonic approximation ((true - harmonic) / true)
# ---

def getRelativeDiff(tru, approx):
    return 100*(tru-approx)/tru

truePeriod = getApproxPeriod(Uthetas)
approxPeriod = getApproxPeriod(sAAThetas)


print(f"The true period of the oscillator is {truePeriod:.2f}s")
```

```python
print(f"The approximate harmonic period of the oscillator is {approxPeriod:.2f}s")
print(f"\nThe relative difference in period is {getRelativeDiff(truePeriod, approxPeriod):.2f}%")


#%%
# Q4 (4 pts) Scanning starting angles over angles from close to 0 to close to $\pi$ radians, plot the period vs the starting angle, and the relative difference with the harmonic approximation vs the starting angle.
# ---

startingAngles = np.delete(np.linspace(0, np.pi, 502), [0, -1]) # Creating a linear space between 0 and pi, then dropping the 0, and pi elements

periods = []
for i in range(len(startingAngles)):
    periods.append(getApproxPeriod(odeint(dU_dt, [startingAngles[i], thetaDot0_rads], ts[:int((4/5*300))])[:, 0]))

periods = np.array(periods)

plt.xlabel("Starting Angle (rad)")
plt.ylabel("Period (s)")
plt.title("Period vs Starting angle")

plt.plot(startingAngles, periods)
plt.show()

plt.xlabel("Starting Angle (rad)")
plt.ylabel("Relative Difference")
plt.title("Accuracy of harmonic approximation")

plt.plot(startingAngles, getRelativeDiff(truePeriod, periods))
plt.show()

# %%
```



Oscillator
2
1
0
−1
−2
θ (rad)
0
5
10
15
20
t (s)


Period vs Starting angle
Period (s)
Starting Angle (rad)
10
15
20
25
0.0
0.5
1.0
1.5
2.0
2.5
3.0


Accuracy of harmonic approximation
Relative Difference
Starting Angle (rad)
0
−50
−100
−150
−200
0.0
0.5
1.0
1.5
2.0
2.5
3.0