Oscillator

May 20, 2020

0.1 Harmonic Oscillators Motion

This code simulates the motion of 15 harmonic oscillators. They are harmonics of three oscillators with slightly different lengths start all moving together. All units are in SI.

[ ]:
```
import numpy as np
from matplotlib import pyplot as plt
import sys

deg2rad = np.pi/180. # degree to radian conversion


class Oscillator :

    def __init__(self, h = 2.5, length = 1., g = 9.807) :
        self._h = h
        self._g = g # gravity acceleration in SI
        self._length = length
        if self._h < self._length :
            print ( "Hanging point should be higher than the length" )
            sys.exit()
        omega = np.sqrt( g/length ) # Computing the oscillation frequency from␣
↪the length
        self.period = round(2*np.pi/omega, 3) # Oscillation Period

    # Defining getter and setter for g
    def get_g( self ) :
        return self._g

    def set_g( self, gg ) :
        if (gg < 0) or ( gg > 20 ) :
            print(  " Not an acceptable physical g value." )
            print( " Keeping the default value g =", self._g )
        else :
            self._g = gg

    g = property( get_g, set_g )
```

```
    # Defining getter and setter for length
    def get_length( self ) :
        return self._length

    def set_length( self, ll ) :
        if ll < 0 :
            print( " Not an acceptable length value." )
            print( " Keeping the default value length =", self._length )

        else :
            self._length = ll

    length = property( get_length, set_length )

    # oscXY method computes the x,y coordinates of the oscillator in the given␣
↪times
    def oscXY( self, **iniconf ) :
        # iniconf is a dictionary with three keys : initial time, final time␣
↪and number of time steps

        g = self._g
        length = self._length
        omega = np.sqrt( g/length ) # oscillation frequency

        t = np.linspace( iniconf["tmin"], iniconf["tmax"], iniconf["step"] )
        theta0 = 6. * deg2rad # initial angle between the oscillator and the␣
↪y-axis (I have used small angle approx)
        theta = theta0 * np.cos( omega*t )
        x = length * np.sin(theta) # x-coordinate
        y = ( self._h - length*np.cos(theta) ) / np.sqrt(2.) # y-coordinate

        return x, y


    # Plotting x,y coordinate in a given time for one occilator
    def makeFig( self, x, y ) :
        for i in range( x.shape[0] ) :
            plt.xlim(-.11, .11)
            plt.ylim(-.005, .02 )
            cc = "blue"
            plt.scatter(x[i], y[i], color=cc)
            if i < 10 :
                plt.savefig("/Users/farhang/Desktop/FigTest/fig{}{}.png".
↪format(0,i),
                            transparent=True, dpi=300, bbox_inches='tight')
            else :
```

```
                plt.savefig("/Users/farhang/Desktop/FigTest/fig{}.png".
↪format(i),
                        transparent=True, dpi=300, bbox_inches='tight')
            plt.show()


    # Plotting x,y coordinates in a given time for multiple oscillators
    def makeFigList( self, X, Y ) :

        # Colour list ( assigne a colour to each oscillator )
        cp = [ "b", "g", "r", "c", "m" ] * 3

        for j in range( X[0].shape[0] ) :
            plt.xlim(-5.*X[0][0], 5.*X[0][0])
            plt.ylim(-.5, self._h )
            for i in range( len(X) ) :
                if i<4 : cc = "b"
                elif i<8 : cc = "r"
                elif i<12 : cc = "g"
                else : cc = "k"
                plt.scatter( X[i][j], Y[i][j], color=cp[i] )

            if j < 10 :
                plt.savefig("/Users/farhang/Desktop/FigTest2/fig{}{}{}.png".
↪format(0,0,j),
                        transparent=True, dpi=300, bbox_inches='tight')

            elif j < 100 :
                plt.savefig("/Users/farhang/Desktop/FigTest2/fig{}{}.png".
↪format(0,j),
                        transparent=True, dpi=300, bbox_inches='tight')

            else :
                plt.savefig("/Users/farhang/Desktop/FigTest2/fig{}.png".
↪format(j),
                        transparent=True, dpi=300, bbox_inches='tight')

            plt.show()
```

[ ]:
```
# Testing the Oscillator class for a single oscillator
myOsc = Oscillator(length=2.)
Tlong = myOsc.period
```

[ ]:

```python
# Multiple oscillators (harmonics of an initial length)
# We consider 13 harmonics of any ocillator :
Tfrac = [  3/4, 3/5, 3/6, 3/7, 5/6, 5/7, 5/8, 5/9, 7/8, 7/9, 7/12, 7/13 ]
Tfrac.append(1)

X = []
Y = []
for frac in Tfrac :
    myOsc = Oscillator( length = 2.*frac*frac )
    x, y = myOsc.oscXY( tmin=0., tmax=105*Tlong, step=1000 )
    X.append(x)
    Y.append(y)

myOsc.makeFigList(X, Y)
```

[ ]:

```python
# Multiple oscillators with 3 different initial lengths
#Tfrac = [  3/4, 3/5, 3/6, 5/6, 5/7, 5/8 ]
Tfrac = [  1./2, 3./4, 6./7, 12./13 ]
Tfrac.append(1.)

lmax = [2.05, 2., 1.95]
X = []
Y = []

for lgt in lmax :
    for frac in Tfrac :
        myOsc = Oscillator( length = lgt*frac*frac )
        #x, y = myOsc.oscXY( tmin=0., tmax=45*Tlong, step=1000 )
        x, y = myOsc.oscXY( tmin=0., tmax=72*Tlong, step=1000 )
        X.append(x)
        Y.append(y)

myOsc.makeFigList(X, Y)
```

[ ]:

```python

```