More on scatter plot animations with Python

The following script shows how to animate a scatter plot. We have seen an example last week with a single point moving up across the plot window, but this time we have many points. It uses some additional functionalities (numpy **`vstack`**) and methods of a scatter plot instance (e.g. **`set_sizes()`** and **`set_facecolor()`**).

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

npoint=500
nframe=100
fig, ax = plt.subplots()
plt.ylim(-0.2,1.2)
plt.xlim(-0.2,1.2)
x = np.random.random(npoint)
y = np.random.random(npoint)
s = 500 * np.random.random(npoint)
c = np.random.random(npoint * 3).reshape(npoint,3)

im = ax.scatter(x,y)
im.set_sizes(s)
im.set_facecolor(c)
dstep=400.

def update_point(num):
    newx=x+np.random.randn(npoint)/dstep
    newy=y+np.random.randn(npoint)/dstep
    data=np.stack((newx,newy),axis=-1)
    im.set_offsets(data)
    return im,

animation = animation.FuncAnimation(fig, update_point,
nframe,interval=1,repeat=False)
```

Project 4 (Due date Thursday November 17th 12 a.m.)

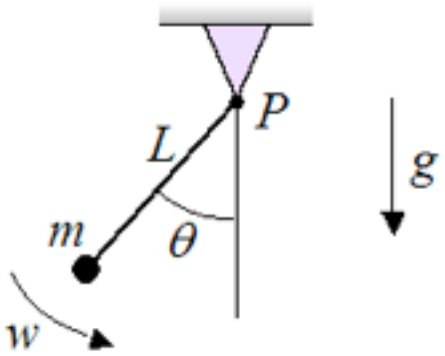
For this project you will study the resonance of a damped and forced single pendulum. Some of you may have studied resonance with LRC (electronic) systems and/or simple

harmonic oscillators. It is ok if you have not. You can (should!) read more about it on this page:

https://en.wikipedia.org/wiki/Mechanical_resonance

Basically, an oscillator has a "natural" frequency, and when the driven force frequency is close to the natural frequency, the system enters is a state of greater amplitude than for any other driven force frequency. In this project you will reproduce this experiment numerically.

Consider a pendulum of mass m attached to a rod of length L.



The angle Θ represents the angle of the rod to the vertical, and g is the acceleration due to gravity. We denote *I* the moment of inertia of the pendulum. The equation of motion is given by:

$$\frac{\mathrm{d}^2\theta}{\mathrm{dt}^2} = -\frac{\mathrm{b}}{I}\frac{\mathrm{d}\theta}{\mathrm{dt}} - \frac{\mathrm{mgL}}{I}\sin(\theta) + \frac{\mathrm{F}}{I}\cos(\omega t)\cos(\theta)$$

In the right -hand-side, the first term is the damping (friction), the second term is the force of gravity and the third is the forced force (note that the cos(Θ) term in the driven force is to ensure that the force cancels out when Θ is +/-π/2). F is the forced force amplitude, ω is its angular frequency and b the damping coefficient. For this problem you will adopt the following values:

g=9.8 $m.s^{-2}$
m=1 kg
L=1 m

Objective: You have to produce a plot of the **maximum** amplitude, max(Θ(t)), as function of the frequency of the forced force frequency ω. The resonance frequency ω_0 is given by:

$$\omega_0 = \sqrt{\frac{\mathrm{g}}{\mathrm{L}}}$$

Your plot has to start well before ω_0 and stop well above it, so we can clearly see what is happening at the resonance frequency. You will have to adopt some values for the other physical quantities of the problem. Here are my suggested values, but feel free to pick up other values if you want:
F=0.1 and b=0.15 (work out the units of F and b)

You can start with initial conditions Θ(t=0)=0 and zero initial velocity. For the moment of inertia, take $I=mL^2$.
You will call this plot **resonance.pdf**. Your script should be called **resonance.py**.
You also have to generate an animation of the pendulum and call this file **pendulum.mp4** (feel free to pick up any values of ω for your animation). Lastly, you will generate a set plots showing Θ(t) and its time derivative dΘ/dt as function of time, for $\omega=\omega_0$, $\omega<<\omega_0$ i and $\omega>>\omega_0$. These plots should be in file **phase_space.pdf**.

Tips: no major difficulties in this project, but you have to make sure you choose your time interval and time steps wisely to make sure you capture the whole behavior of the oscillator.