PHY224H1F

Exercise 2: *The Pendulum (II)*

2.1 Numerical Integrators

In the previous exercise, we analyzed the equation of motion of a simple pendulum and wrote the Python code needed to numerically integrate the equation. If you strictly used the numerical approximation from Exercise 1, eq. (7), you must have found that everything failed catastrophically: angle amplitudes increased and energy was not conserved.

The reason was the integration method. We used the most primitive numerical integration method, called **Euler Forward** (explicit):

$$\begin{aligned} \theta[i+1] &= \theta[i] + \omega[i]\Delta t \\ \omega[i+1] &= \omega[i] - \Omega_0^2\theta[i]\Delta t \end{aligned} \qquad (1)$$

The method increases the angle through an interval Δt using derivative information from only the beginning of the interval. **It can be proved (see Reference at the end) that the method is unstable which means that oscillation amplitude as well as total energy monotonically increases in time**. The numerical solution has a spiral orbit in the (θ, ω) phase space.

The method's accuracy and stability can be improved by decreasing the time step, which makes it attractive because of its simplicity of implementation.

Similar to (1), the **Euler Backward Method** (implicit) is given by:

$$\begin{aligned} \theta[i+1] &= \theta[i] + \omega[i+1]\Delta t \\ \omega[i+1] &= \omega[i] - \Omega_0^2\theta[i+1]\Delta t \end{aligned} \qquad (1')$$

The method is implicit because both $\theta[i+1]$, $\omega[i+1]$ are used on the right hand side; it is stable and therefore allows large time steps to be taken. However, it involves some numerical dissipation (see Reference).

A very simple remedy is to combine the two Euler methods into:

$$\begin{aligned} \theta[i+1] &= \theta[i] + \omega[i]\Delta t \\ \omega[i+1] &= \omega[i] - \Omega_0^2\theta[i+1]\Delta t \end{aligned} \qquad (2)$$

The resultant is an explicit method, stable, with no numerical dissipation, called **Euler-Cromer** or **Symplectic Euler Method (SEM*).***

➔ Insert the new code lines from SEM into the program you wrote last time and plot: 'Energy vs. time' and the phase plot. What happens?

➔ Compare and discuss the phase plots from: Forward Euler, and Symplectic Method. Use the same time step in all.

We have to keep in mind that in integrating conservative problems is it essential to use a symplectic method.

2.2 Pendulum at large angles

Write the Python code to integrate the equation of motion of the pendulum at large angle. What you have to change are the following:

- initial angle (in radians)
- equations of motion become:

$$\frac{d\omega}{dt} = -\Omega_0^2 \sin\theta \quad and \quad \frac{d\theta}{dt} = \omega$$

- numerical solution has to be re-written
- energy expression is now: $E = \frac{1}{2} mL^2\omega^2 + mgL(1-\cos\theta)$

➔ Use the modified code template from Exercise 1 with SEM and change it to account for the large oscillation angle. Extend the time to 2 minutes. Plot 'Angle vs. time', 'Energy vs. time' and the phase plot.

2.3 Adding a damping term.

There is a large discrepancy between the 'angle vs. time' Python plot and the real experimental data (open the RMS.vi application and let the pendulum swing for a couple of minutes).

It is obvious that the equations of motion you used so far modeled the pendulum *without damping (physical dissipation).* In general, the damping force exerted on a body moving in air or water depends on velocity $\vec{v}$ of the body relative to the medium according to:

$$\vec{F}_d = -\frac{1}{2} C\rho A\vec{v}|\vec{v}| \qquad (3)$$

Coefficients used in (3) are:

C = drag coefficient (dimensionless)
A = cross-sectional area perpendicular to the flow (m^2)
ρ = density of the medium (kg/m^3)
v = linear velocity of the body relative to the medium (m/s)

The direction of the damping (drag) force is always opposite to the direction of velocity. The drag coefficient C is not constant: C depends on body velocity, but also on viscosity of the medium, the shape of the body, and the roughness of its surface.

The **Reynolds number R_e** has been found to be a useful dimensionless quantity that characterizes the dependence of the drag coefficient on velocity.
The Reynolds number is the ratio of the inertial force of the medium to the viscous force:

$$R_e = \frac{\ell\rho v}{\eta} \qquad (4)$$

R_e = Reynolds number (dimensionless)
ℓ = Characteristic length of the body along the direction of flow (m)
η = Dynamic viscosity of the medium (N s/m^2)

ρ = Density of the medium (kg/m^3)
v = Linear velocity of the body relative to the medium (m/s)

If the flow is *laminar*, the Reynolds number takes small values (R_e <2300) and the drag coefficient is inversely proportional to the velocity. This makes the drag force directly proportional to the velocity: $\vec{F}_d \approx -\vec{v}$

When the flow is *turbulent*, the Reynolds number is large (4000 < R_e) the drag coefficient is approximately constant and the drag force is dependent on the square of the velocity.

➔Calculate the Reynolds number for the large angle pendulum.

Motion and speed of a typical pendulum bob in air at large oscillation angles correspond to small Reynolds numbers. Therefore, the equation of motion of our pendulum would be written as:

$$\frac{d^2\theta}{dt^2} + \Omega_0^2 \sin\theta + \gamma\left(\frac{d\theta}{dt}\right) = 0 \qquad (5)$$

The damping term was written to include the linear velocity: $v = L\omega = L\frac{d\theta}{dt}$.

➔Using cursors, take 5-6 readings of amplitude. Assuming an exponential decay of the oscillation *envelope*, estimate the decay constant γ (γ/2 is the inverse of time for which amplitude falls to 1/e of the initial value). You do not have to use Python to do this calculation.

2.4 The Python application

The coupled equations we need to formulate for our next Python application are:

$$\frac{d\omega}{dt} = -\Omega_0^2 \sin\theta - \gamma\omega \qquad (6)$$
$$\frac{d\theta}{dt} = \omega$$

➔ Use the template from part 2.2, modify it according to (6) and plot 'Angle vs. time'. Discuss the plot.

2.5 Compare with experimental data (qualitatively)

Open the RMS.vi application. Remember that the initial position of the rotational motion sensor sets up the origin of the vertical axis.

Take the pendulum out of equilibrium by 30^o and let it swing for as long as you need in order to see a significant decay in amplitude. **Qualitatively** compare the experimental data with the output of your program. What do you think makes them different?

*Note. Comparison with some experimental data was justified only for qualitative purposes .We have not attempted to **fit** the data. This will be the topic of the next exercise.*

All requirements marked by (➔) have to be submitted to your TA.

Reference: W.H. Press, S.A. Teukolsky, W.T. Vetterling & B.P. Flannery (1992), Numerical Recipes in C: the art of scientific computing, 2nd ed. Cambridge University Press.

Written by Ruxandra Serbanescu (v.5, 2009-2012)