ps4_solutions

October 28, 2019

```
In [1]: from __future__ import division
        import sympy
        from sympy import *
        from sympy import init_printing
        from sympy.physics.vector import dynamicsymbols
        init_printing()
```

Problem 1. Solve problem 18 from Goldstein Chapter 2 (third edition): A point mass is constrained to move on a massless hoop of radius a fixed in a vertical plane that rotates about its vertical symmetry axis with constant angular speed ω.

(a) Obtain the Lagrange equations of motion assuming the only external forces arise from gravity.

(b) Show that the energy of the mass, given by the naive expression, E = T + V, is not conserved. Give a qualitative explanation why.

(c) What are the constants of motion?

(d) Show that if ω is greater than a critical value ω_0, there can be a solution in which the particle remains stationary on the hoop at a point other than the bottom, but that if $\omega < \omega_0$, the only stationary point for the particle is at the bottom of the hoop.

(e) What is the value of ω_0?

```
In [2]: g, m, a, x, y, V, T, omega = symbols('g m a x y V T \omega')
        t = symbols("t")
        theta = dynamicsymbols(r'\theta')
        thetadot = diff(theta,t)
        thetaddot = diff(thetadot,t)
        #auxiliary symbols for prettier printing
        theta_symbol = symbols(r'\theta')
        thetadot_symbol = symbols(r'\dot\theta')
        thetaddot_symbol = symbols(r'\ddot\theta')
        alias = {thetaddot: thetaddot_symbol,
                 thetadot: thetadot_symbol,
                 theta: theta_symbol}
```

```
In [3]: T = m*a**2*thetadot**2/2 + m*omega**2*a**2*sin(theta)**2/2
        T.subs(alias)
```

Out[3]:

$$\frac{\dot{\theta}^2 m}{2} a^2 + \frac{\omega^2 m}{2} a^2 \sin^2(\theta)$$

In [4]:
```
V = -m*g*a*cos(theta)
V.subs(alias)
```

Out[4]:

$$-agm\cos(\theta)$$

In [5]:
```
L = T-V
L.subs(alias)
```

Out[5]:

$$\frac{\dot{\theta}^2 m}{2} a^2 + \frac{\omega^2 m}{2} a^2 \sin^2(\theta) + agm\cos(\theta)$$

In [6]:
```
eom1 = diff(diff(L,thetadot),t)-diff(L,theta)
eom1 = simplify(eom1)
relational.Eq(eom1.subs(alias))
```

Out[6]:

$$am\left(\ddot{\theta}a - \frac{\omega^2 a}{2}\sin(2\theta) + g\sin(\theta)\right) = 0$$

In [7]:
```
thetaddot_solution = solve(eom1,thetaddot)[0]
relational.Eq(thetaddot_solution)
```

Out[7]:

$$\frac{1}{a}\left(\omega^2 a\cos(\theta(t)) - g\right)\sin(\theta(t)) = 0$$

In [8]:
```
solve(thetaddot_solution,theta)
```

Out[8]:

$$\left[0,\quad \pi,\quad -\operatorname{acos}\left(\frac{g}{\omega^2 a}\right) + 2\pi,\quad \operatorname{acos}\left(\frac{g}{\omega^2 a}\right)\right]$$

Thus, four equilibrium solutions ($\dot{\theta} = \ddot{\theta} = 0$) are possible. The first one is trivial, $\theta(t) = 0$, i.e., the pendulum is at the bottom of the hoop. The second one is an unstable solution, $\theta(t) = \pi$, which represents the particle at the top of the hoop. The third and fourth solutions are non-trivial and stable. They are equivalent to each other (up to a sign in θ) and correspond to:

In [9]:
```
relational.Eq(cos(theta),solve(thetaddot_solution,cos(theta))[0])
```

Out[9]:

$$\cos\left(\theta(t)\right) = \frac{g}{\omega^2 a}$$

Clearly, this non-trivial solution is physical if and only if $g \leq \omega^2 a$, or $\omega \geq \omega_0 = \sqrt{g/a}$. Thus, if $\omega < \omega_0$, the only solution is at the bottom of the hoop, $\theta = 0$. If $\omega = \omega_0$ the non-trivial and trivial solutions coincide.

WHAT KEEPS OMEGA CONSTANT?

Note that the Euler-Lagrange equation in the φ-direction is not very useful: that's because the ring would not rotate at a constant angular velocity ω on its own: in fact, you'd need to apply a torque to guarantee $\omega =$ constant! In fact, till now, we have not even introduced the φ-coordinate. In the below, all we need to know is that the Lagrantian is independent of φ (i.e., $\partial L/\partial\varphi = 0$) and that $\dot{\varphi} = \omega$. Thus, the equations in the φ-direction take the form:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\varphi}}\right) + \frac{\partial L}{\partial \varphi} = Q_\varphi, \tag{1}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \omega}\right) = Q_\varphi, \tag{2}$$

$$\frac{d}{dt}(\omega m a^2 \sin^2\theta) = Q_\varphi, \tag{3}$$

$$\omega m a^2 2 \sin\theta\cos\theta\,\dot{\theta} = Q_\varphi, \tag{4}$$

where the latter equation holds because $\omega =$ constant by the problem setup. In fact, this equation gives us the expression for Q_φ, the generalized force in the φ-direction that ensures that the ring maintains a constant angular velocity, as required by the problem. This force is non-zero only when the bead moves along the rotating ring (i.e., when both $\dot{\theta}$ and ω are non-zero).

ENERGY NON-CONSERVATION

Because this force does work on the system, the total energy is NOT conserved! In other words, in order to maintain $\omega =$ constant, we have to attach an motor to the ring that either speeds it up if the bead moves away from the axis ($\dot{\theta} > 0$) or slow it down if the bead moves toward the axis ($\dot{\theta} < 0$). So, is there a quantity that is conserved instead of energy? Could ω be that conserved quantity? Even though it is a constant, it is not a conserved quantity because it is constant only because of the particular choice of Q_φ by the problem creators. Its constancy is not due to a symmetry of the system. Instead, it is enforced externally by the hard-working motor.

Is there another, intrinsic conserved quantity? Yes! To find it, we need to construct the *Hamiltonian*, as we discussed in class. First, we compute the momentum associated with the θ direction:

In [10]:
```
#theta-momentum
ptheta = L.diff(thetadot)
ptheta.subs(alias)
```

Out[10]:

$$\dot{\theta} a^2 m$$

Now, we are in a position to compute the Hamlitonian. Note that we canNOT write the Hamiltonian as the sum of T and V because V is not of the simple bilinar form. Hence, we have to resort to the more general expression:

$$H = \sum_i p_i \dot{q}_i - L$$

In [11]:
```
#Hamiltonian, Goldstein eqs. (2.53) and (8.41)
H = ptheta * thetadot - L
relational.Eq(Symbol("H"),H.subs(alias))
```

Out[11]:

$$H = \frac{\dot{\theta}^2 m}{2} a^2 - \frac{\omega^2 m}{2} a^2 \sin^2(\theta) - agm\cos(\theta)$$

Because the Lagrangian is time-independent, Hamiltonian is conserved ($dH/dt = 0$). Thus, H is the sought conserved quantity.

Problem 2. There are two systems sketched below. For each system, do the following:

(a) Find the matrices $\boldsymbol{T}$ and $\boldsymbol{V}$ such that $L = 0.5 T_{ij}\dot{\eta}_i\dot{\eta}_j - 0.5 V_{ij}\eta_i\eta_j$. Be sure to say what η_i represents, and also what the equilibrium position is.

(b) Find the eigenfrequencies.

(c) Find the matrix of eigenvectors $\boldsymbol{A}$, properly orthogonalized and normalized (i.e., such that $\tilde{\boldsymbol{A}}\boldsymbol{T}\boldsymbol{A} = \mathbf{1}$).

(d) Describe physically the behavior of each of the modes, with sketches if necessary.

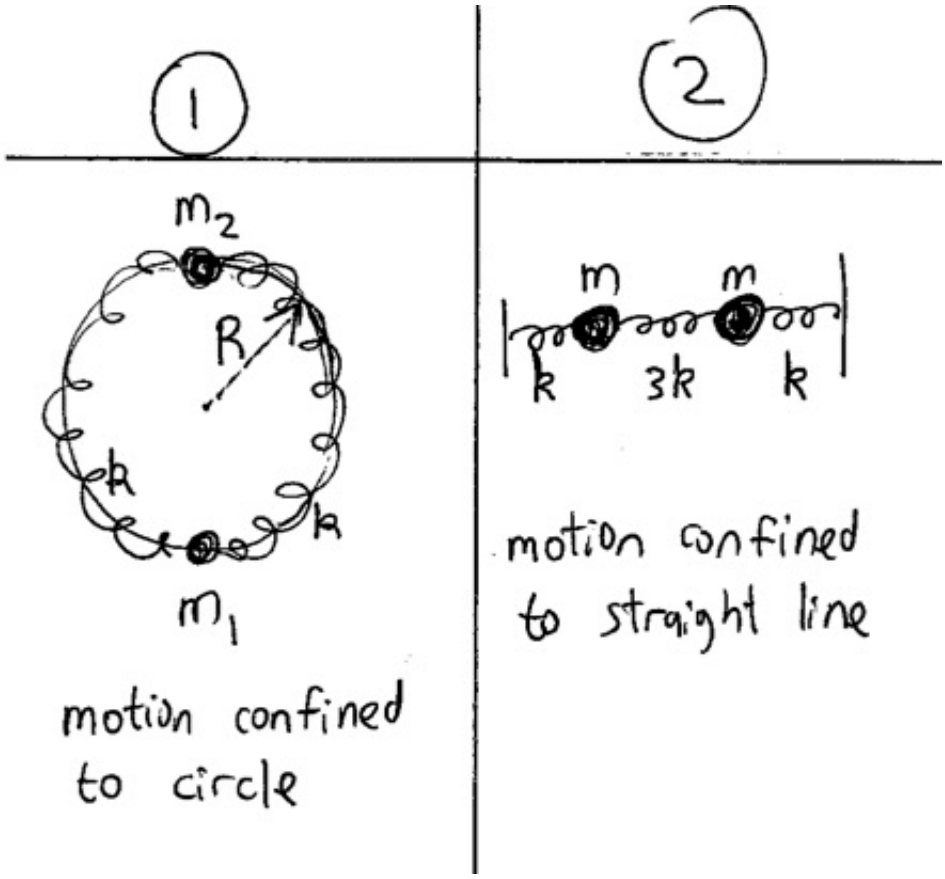
System (1):

In [12]:
```
R = symbols("R",positive=True)
T, L, V = symbols(r'T L V')
m, m1, m2, mu, k = symbols(r'm m_1 m_2 \mu k',positive=True)
t = symbols("t")
phi1 = dynamicsymbols(r'\varphi_1')
phi1dot = diff(phi1,t)
phi1ddot = diff(phi1dot,t)
phi1_symbol = symbols(r'\varphi_1')
phi1dot_symbol = symbols(r'\dot\varphi_1')
phi1ddot_symbol = symbols(r'\ddot\varphi_1')
phi2 = dynamicsymbols(r'\varphi_2')
phi2dot = diff(phi2,t)
phi2ddot = diff(phi2dot,t)
phi2_symbol = symbols(r'\varphi_2')
phi2dot_symbol = symbols(r'\dot\varphi_2')
phi2ddot_symbol = symbols(r'\ddot\varphi_2')
alias = {phi1ddot: phi1ddot_symbol, phi2ddot: phi2ddot_symbol,
         phi1dot: phi1dot_symbol, phi2dot: phi2dot_symbol,
         phi1: phi1_symbol, phi2: phi2_symbol, m1*m2/(m1+m2): mu}
```

In [13]:
```
T = m1*(R*phi1dot)**2/2 + m2*(R*phi2dot)**2/2
T.subs(alias)
```

Out[13]:

$$\frac{\dot{\varphi}_1^2 m_1}{2} R^2 + \frac{\dot{\varphi}_2^2 m_2}{2} R^2$$


1
2
m_2
R
k
k
m_1
motion confined
to circle
m
m
k
3k
k
motion confined
to straight line

title

```
In [14]: V = k*(R*(phi2-phi1))**2/2 + k*(R*(2*pi-phi2+phi1))**2/2
         V.subs(alias)
```

Out[14]:

$$\frac{R^2k}{2}\left(-\varphi_1+\varphi_2\right)^2+\frac{R^2k}{2}\left(\varphi_1-\varphi_2+2\pi\right)^2$$

```
In [15]: Tij = Matrix([[diff(T,phi1dot,phi1dot), diff(T,phi1dot,phi2dot)],
                       [diff(T,phi2dot,phi1dot), diff(T,phi2dot,phi2dot)]])
         Tij
```

Out[15]:

$$\begin{bmatrix} R^2m_1 & 0 \\ 0 & R^2m_2 \end{bmatrix}$$

```
In [16]: Vij = Matrix([[diff(V,phi1,phi1), diff(V,phi1,phi2)],
                       [diff(V,phi2,phi1), diff(V,phi2,phi2)]])
         Vij
```

Out[16]:

$$\begin{bmatrix} 2R^2k & -2R^2k \\ -2R^2k & 2R^2k \end{bmatrix}$$

```
In [17]: x = symbols("x")
         matrix = -x*Tij+Vij
         determinant = matrix.det()
         determinant
```

Out[17]:

$$-4R^4k^2+\left(2R^2k-R^2m_1x\right)\left(2R^2k-R^2m_2x\right)$$

```
In [18]: evals=solve(determinant.subs(alias),x)
         evals
```

Out[18]:

$$\left[0,\quad \frac{2k}{m_1m_2}\left(m_1+m_2\right)\right]$$

This gives us the eigenfrequencies, $\omega_1^2 = 0$ and $\omega_2^2 = 2k/\mu$, where μ is the reduced mass, $\mu = m_1m_2/(m_1+m_2)$.

```
In [19]: a, b = symbols("a b")
         evec = Matrix([a,b])

In [20]: matrix.subs(x,evals[0])
```

Out[20]:

$$\begin{bmatrix} 2R^2k & -2R^2k \\ -2R^2k & 2R^2k \end{bmatrix}$$

In [21]:
```
sol1 = solve(matrix.subs(x,evals[0])*evec,a,b)
evec1 = evec.subs(sol1)
evec1
```

Out[21]:

$$\begin{bmatrix} b \\ b \end{bmatrix}$$

In [22]:
```
#normalize eigenvector 1
bsol1 = solve((evec1.T*Tij*evec1)[0]-1,b)[1]
evec1sol = evec1.subs(b,bsol1)
evec1sol
```

Out[22]:

$$\begin{bmatrix} \frac{1}{R\sqrt{m_1+m_2}} \\ \frac{1}{R\sqrt{m_1+m_2}} \end{bmatrix}$$

In [23]:
```
sol2 = solve(matrix.subs(x,evals[1])*evec,a,b)
evec2 = evec.subs(sol2)
evec2
```

Out[23]:

$$\begin{bmatrix} -\frac{bm_2}{m_1} \\ b \end{bmatrix}$$

In [24]:
```
#normalize eigenvector 2
bsol2 = solve((evec2.T*Tij*evec2)[0]-1,b)[1]
evec2sol = evec2.subs(b,bsol2)
evec2sol
```

Out[24]:

$$\begin{bmatrix} -\frac{\sqrt{m_2}}{R\sqrt{m_1}\sqrt{m_1+m_2}} \\ \frac{\sqrt{m_1}}{R\sqrt{m_2}\sqrt{m_1+m_2}} \end{bmatrix}$$

In [25]:
```
#combine the eigenvectors into matrix
Aij = Matrix([evec1sol.T,evec2sol.T]).T
Aij
```

Out[25]:

$$\begin{bmatrix} \frac{1}{R\sqrt{m_1+m_2}} & -\frac{\sqrt{m_2}}{R\sqrt{m_1}\sqrt{m_1+m_2}} \\ \frac{1}{R\sqrt{m_1+m_2}} & \frac{\sqrt{m_1}}{R\sqrt{m_2}\sqrt{m_1+m_2}} \end{bmatrix}$$

The first eigenvector (the left column of the matrix) is rotation, and the second eigen vector is oscillation toward and away from each other, with the amplitude inversely proportional to the mass. We can now verify that the eigenvectors are orthonormal:

```
In [26]: #test that the eigenvectors are orthonormal
         simplify(Aij.T*Tij*Aij)
```

Out[26]:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

System (2):

```
In [27]: R = symbols("R",positive=True)
         m1, m2, T, L, V, k, k1, k2, k3, l = symbols('m_1 m_2 T L V k k_1 k_2 k_3 l')
         m = symbols("m", positive=True)
         t = symbols("t")
         x1 = dynamicsymbols(r'x_1')
         x1dot = diff(x1,t)
         x1ddot = diff(x1dot,t)
         x1_symbol = symbols(r'x_1')
         x1dot_symbol = symbols(r'\dot{x_1}')
         x1ddot_symbol = symbols(r'\ddot{x_1}')
         x2 = dynamicsymbols(r'x_2')
         x2dot = diff(x2,t)
         x2ddot = diff(x2dot,t)
         x2_symbol = symbols(r'x_2')
         x2dot_symbol = symbols(r'\dot{x_2}')
         x2ddot_symbol = symbols(r'\ddot{x_2}')
         alias = {x1ddot: x1ddot_symbol, x2ddot: x2ddot_symbol,
                  x1dot: x1dot_symbol, x2dot: x2dot_symbol,
                  x1: x1_symbol, x2: x2_symbol}
```

```
In [28]: T = m*x1dot**2/2 + m*x2dot**2/2
         T.subs(alias)
```

Out[28]:

$$\frac{\dot{x_1}^2 m}{2} + \frac{\dot{x_2}^2 m}{2}$$

```
In [29]: V = k*x1**2/2 + 3*k*(x2-x1)**2/2 + k*(l-x2)**2/2
         V.subs(alias)
```

Out[29]:

$$\frac{kx_1^2}{2} + \frac{k}{2}(l - x_2)^2 + \frac{3k}{2}(-x_1 + x_2)^2$$

In [30]:
```
Tij = Matrix([[diff(T,x1dot,x1dot), diff(T,x1dot,x2dot)],
              [diff(T,x2dot,x1dot), diff(T,x2dot,x2dot)]])
Tij
```

Out[30]:

$$\begin{bmatrix} m & 0 \\ 0 & m \end{bmatrix}$$

In [31]:
```
Vij = Matrix([[diff(V,x1,x1), diff(V,x1,x2)],
              [diff(V,x2,x1), diff(V,x2,x2)]])
Vij
```

Out[31]:

$$\begin{bmatrix} 4k & -3k \\ -3k & 4k \end{bmatrix}$$

In [32]:
```
#x = \omega^2
x = symbols("x")
matrix = -x*Tij+Vij
determinant = matrix.det()
factor(determinant)
```

Out[32]:

$$(-7k + mx)(-k + mx)$$

In [33]:
```
evals=solve(determinant,x)
evals
```

Out[33]:

$$\left[\frac{k}{m}, \quad \frac{7k}{m}\right]$$

Thus, $\omega_1 = (k/m)^{1/2}$ and $\omega_2 = (7k/m)^{1/2}$.

In [34]:
```
a, b = symbols("a b")
evec = Matrix([a,b])
```

In [35]:
```
matrix.subs(x,evals[0])
```

Out[35]:

$$\begin{bmatrix} 3k & -3k \\ -3k & 3k \end{bmatrix}$$

```
In [36]: sol1 = solve(matrix.subs(x,evals[0])*evec,a,b)
         evec1 = evec.subs(sol1)
         evec1
```

Out[36]:

$$\begin{bmatrix} b \\ b \end{bmatrix}$$

```
In [37]: #normalize eigenvector 1
         bsol1 = solve((evec1.T*Tij*evec1)[0]-1,b)[1]
         evec1sol = evec1.subs(b,bsol1)
         evec1sol
```

Out[37]:

$$\begin{bmatrix} \frac{\sqrt{2}}{2\sqrt{m}} \\ \frac{\sqrt{2}}{2\sqrt{m}} \end{bmatrix}$$

```
In [38]: sol2 = solve(matrix.subs(x,evals[1])*evec,a,b)
         evec2 = evec.subs(sol2)
         evec2
```

Out[38]:

$$\begin{bmatrix} -b \\ b \end{bmatrix}$$

```
In [39]: #normalize eigenvector 2
         bsol2 = solve((evec2.T*Tij*evec2)[0]-1,b)[1]
         evec2sol = evec2.subs(b,bsol2)
         simplify(evec2sol)
```

Out[39]:

$$\begin{bmatrix} -\frac{\sqrt{2}}{2\sqrt{m}} \\ \frac{\sqrt{2}}{2\sqrt{m}} \end{bmatrix}$$

```
In [40]: #combine the eigenvectors into matrix
         Aij = Matrix([evec1sol.T,evec2sol.T]).T
         Aij
```

Out[40]:

$$\begin{bmatrix} \frac{\sqrt{2}}{2\sqrt{m}} & -\frac{\sqrt{2}}{2\sqrt{m}} \\ \frac{\sqrt{2}}{2\sqrt{m}} & \frac{\sqrt{2}}{2\sqrt{m}} \end{bmatrix}$$

The first eigenvector is two masses oscillating in phase (left and right together, with the distance between them not changing), and the second out of phase (towards and away from each other, with the distance between them changing).

We can verify that the eigenvectors are orthonormal:

In [41]:
```
#test that the eigenvectors are orthonormal
simplify(Aij.T*Tij*Aij)
```

Out[41]:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$