Pêndulo simples

1. Aproximação linear: $\theta \ll 1$

1.2 Formalismo Hamiltoniano

Energias cinética e potencial:

$$K = \frac{1}{2}I\omega^2 = \frac{1}{2}mL^2\omega^2 = \frac{1}{2}mL^2\left(\frac{d\theta}{dt}\right)^2$$
$$U = mgL(1-\cos(\theta)) = mgL\left[1-\left(1-\frac{\theta^2}{2}+\cdots\right)\right] \approx mgL\frac{\theta^2}{2}$$

Lagrangeana:

$$\mathcal{L}(\theta,\dot{\theta}) = K(\dot{\theta}) - U(\theta)$$

In [1]:
```
from matplotlib import pyplot as plt
import numpy as np
from sympy import *
from IPython.display import display

theta = Function('theta')

t = Symbol('t', real="True")

m, g, L, w0, I0 = symbols('m g L omega_0 I_0', positive="True")

K = (m*L**2)*(theta(t).diff(t))**2/2   # Energia Cinética
U =  m*g*L*(theta(t)**2/2)             # Energia Potencial

Lagrangeana = K - U

display(Lagrangeana)
```

$$\frac{L^2 m\left(\frac{d}{dt}\theta(t)\right)^2}{2} - \frac{Lg m\theta^2(t)}{2}$$

Hamiltoniana:

$$\mathcal{H}(\theta,\dot{\theta}) = K(\dot{\theta}) + U(\theta)$$

Momentum:

$$p_\theta = \frac{\partial\mathcal{L}}{\partial\dot{\theta}}$$

In [2]:
```
_Ham = K + U     # Hamiltoniana nao está nas variáveis corretas

v = symbols('v') # velocidade (uso temporario)

q = Function('q')
p = Function('p')

# Lagrangeana e Hamiltoniana em função de v (não dtheta/dt) - simplifica expressões abaixo
_Lag_v  = Lagrangeana.subs({diff(theta(t),t):v})
#display(_Lag_v)
_Ham_v  = _Ham.subs({diff(theta(t),t):v})
#display(_Ham_v)

# p = dL/dv, isolando v = f(p)
sol_v   = solve(Eq(p(t),diff(_Lag_v,v)),v)
#display(sol_v[0])

# Substituir (v -> p) e (theta -> q) na Hamiltoniana
Hamiltoniana = _Ham_v.subs({v:sol_v[0],theta(t):q(t)})
display(Hamiltoniana)

# Substituindo w0^=g/L ou g = L*w0^2 e I0 = mL^2
Hamiltoniana = Hamiltoniana.subs({m:I0/L**2,g:L*w0**2})
```

$$\frac{Lgmq^2(t)}{2} + \frac{p^2(t)}{2L^2m}$$

Equações de Hamilton:

$$\frac{d\theta}{dt} = \frac{d\mathcal{H}}{dp_\theta}$$
$$\frac{dp_\theta}{dt} = -\frac{d\mathcal{H}}{d\theta}$$

In [3]:
```
eqs = (Eq(Derivative(q(t),t),diff(Hamiltoniana,p(t))),
       Eq(Derivative(p(t),t),-diff(Hamiltoniana,q(t))))
display(eqs[0],eqs[1])
```

$$\frac{d}{dt}q(t) = \frac{p(t)}{I_0}$$

$$\frac{d}{dt}p(t) = -I_0\omega_0^2 q(t)$$

In [4]:
```
sol = dsolve(eqs)
display(sol[0],sol[1])
```

$$q(t) = \frac{C_1 \cos(\omega_0 t)}{I_0} + \frac{C_2 \sin(\omega_0 t)}{I_0}$$

$$p(t) = -C_1\omega_0 \sin(\omega_0 t) + C_2\omega_0 \cos(\omega_0 t)$$

In [5]:
```
func_x = sol[0].rhs
func_p = sol[1].rhs

x0, p0 = symbols('x_0 p_0', real="True")

eqs  = (Eq((func_x.subs({t:0})),x0), Eq(func_p.subs({t:0}),p0))
display(solve(eqs))
```

```
[{C1: I_0*x_0, C2: p_0/omega_0}]
```

In [6]:
```
# Constantes em função das condições iniciais
init_conditions = solve(eqs)[0]
display(init_conditions)

# Substituind0 condições iniciais na solução:
expressao_x = func_x.subs(init_conditions)
expressao_p = func_p.subs(init_conditions)
display(expressao_x,expressao_p)
```

```
{C1: I_0*x_0, C2: p_0/omega_0}
```

$$x_0 \cos(\omega_0 t) + \frac{p_0 \sin(\omega_0 t)}{I_0 \omega_0}$$

$$-I_0 \omega_0 x_0 \sin(\omega_0 t) + p_0 \cos(\omega_0 t)$$

In [7]:
```
# Transformação da solução para uso com valores numéricos
ang = lambdify([(t, x0, p0, w0, I0)], expressao_x)
mom = lambdify([(t, x0, p0, w0, I0)], expressao_p)
```

In [8]:
```
m    = 1
L    = 2
x0   = 3
v0   = 4
tmax = 10

g    = 9.8
w0   = np.sqrt(g/L)
I0   = m*L**2
p0   = I0*v0

tempo = np.linspace(0, tmax, 100) #np.arange(0,tmax,.1)

y1 = ang([tempo,x0,p0,w0,I0])
y2 = mom([tempo,x0,p0,w0,I0])

# make figure
fig, ax = plt.subplots(1,2,gridspec_kw={'width_ratios': [3, 1]}) #,subplot_kw=dict(aspect='e
qual'))
# gridspec_kw={   'width_ratios': [2, 1],                        'height_ratios': [1, 2]}
ax[0].plot(tempo, y1)
ax[0].plot(tempo, y2)
ax[0].set_xlabel('t')
ax[0].set_ylabel('x(t),p(t)')

ax[1].set_xlabel('x')
ax[1].set_ylabel('p')
ax[1].plot(y1,y2)

fig.tight_layout()
plt.show()
```