2020

Pendulum

S.Matoike

目次

第 1 章

単振子

1.1　モデルの定式化

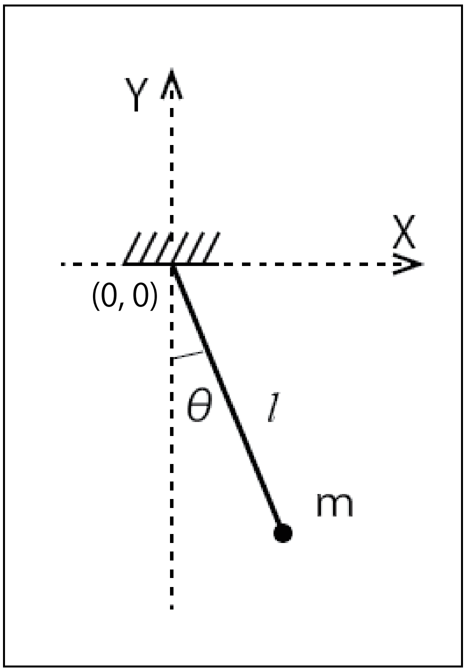

図 1.1　単振子

一方の端が固定され, 質量を無視できる長さ l の棒の他端に, 質量 m の質点が拘束されている. 棒が鉛直となす角は θ, 重力加速度は g である.

$$T = \frac{1}{2}m(l\dot{\theta})^2, \quad U = -mgl\cos\theta$$

より,Lagrangian$L = T - U$ は,

$$L = \frac{1}{2}ml^2\dot{\theta}^2 + mgl\cos\theta$$

Euler-Lagrange eq. は、

$$\frac{\mathrm{d}}{dt}\frac{\partial L}{\partial\dot{\theta}} - \frac{\partial L}{\partial\theta} = 0$$

により次の通り.

$$\ddot{\theta} = -\frac{g}{l}\sin\theta$$

1.2　Python で模擬実験

これを次のように 1 階まで分解し, 連立にして関数 ode を定義する.

$$\frac{\mathrm{d}\theta}{\mathrm{d}t} = \dot{\theta}$$

$$\frac{\mathrm{d}\dot{\theta}}{\mathrm{d}t} = -\frac{g}{l}\sin\theta$$

常微分方程式（Ordinary Differential Equation;ODE）の積分は,scipy.integrate の odeint を使う. 関数 ode の戻り値は,$[\theta, \dot{\theta}]$ である.Python のアニメーションには,FuncAnimation を使う.

ソースコード 1.1　単振子

```python
import matplotlib.pyplot as plt
import numpy as np
from numpy import sin, cos, pi
from matplotlib.animation import FuncAnimation
from scipy.integrate import odeint
#
# Constants
#
G = 9.8                  # [m/s^2] acceleration of gravity
THETA0 = pi/4.       # [rad]   initial angle
V0 = 1.          # [m/s]   initial velocity
L = 1.           # [m]     length of the pendulum
DURATION = 10.   # [s]     duration time
INTERVAL = 0.05  # [s]     interval time
#
# Differential Equation
#
def ode(f, t):
    theta, dtheta = f
    dfdt = [dtheta, -(G/L) * sin(theta)]
    return dfdt
#
# Initial condition
#
f0 = [THETA0, V0/L]     # [theta, v] at t = 0
t = np.arange(0, DURATION + INTERVAL, INTERVAL)     # domain of definition
#
# Solve the equation
#
sol = odeint(ode, f0, t)
```

```python
theta = sol[:, 0]
x = L * sin(theta)
y = - L * cos(theta)   # coordinates of the mass point
#
# Prepare the Screen to display
#
fig = plt.figure()
ax = fig.add_subplot(111, aspect='equal', autoscale_on=False,
                     xlim=(-L, L), ylim=(-L, L))
ax.grid()
markers_on = [1]
line, = plt.plot([], [], 'ro-', markevery=markers_on, animated=True)
time_template = 'time = %.1fs'
time_text = ax.text(0.05, 0.9, '', transform=ax.transAxes)
#
# Animate the simulated results
#
def init():
    time_text.set_text('')
    return line, time_text

def update(i):
    next_x = [0, x[i]]
    next_y = [0, y[i]]
    line.set_data(next_x, next_y)
    time_text.set_text(time_template % (i*INTERVAL))
    return line, time_text

FRAME_INTERVAL = 1000 * INTERVAL # [msec] interval between frames
ani = FuncAnimation(fig, update, frames=np.arange(0, len(t)),
                    interval=FRAME_INTERVAL, init_func=init, blit=True)
#
# Show on the screen and Save the results
#
plt.show()
FPS = 1000/FRAME_INTERVAL # frames per second
#ani.save('single_pendulum.mp4', fps=FPS, extra_args=['-vcodec', 'libx264'])
ani.save('single_pendulum.gif', writer='imagemagick', fps=FPS)
```

第 2 章

2 重平面振子

2.1　モデルの定式化

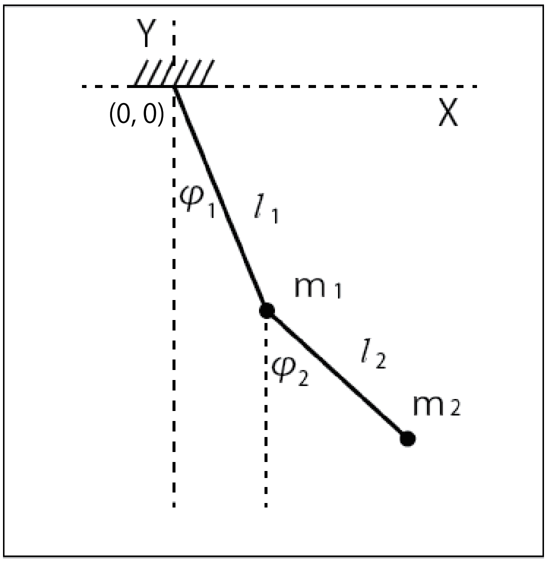

図 2.1　2 重平面振子

棒 l_1 及び l_2 が鉛直となす角を φ_1, φ_2 とする. 座標原点を支点におくことにすると, 質点の座標はそれぞれ

$$(l_1 \sin\varphi_1, -l_1\cos\varphi_1), \qquad\qquad (l_1\sin\varphi_1 + l_2\sin\varphi_2, -l_1\cos\varphi_1 - l_2\cos\varphi_2)$$

質点 m_1 に対する運動エネルギー T_1, 及びポテンシャル・エネルギー U_1 は,

$$T_1 = \frac{1}{2}m_1(l_1\dot{\varphi}_1)^2 \quad , \qquad U_1 = -m_1 g l_1 \cos\varphi_1$$

$(T_1 = \dfrac{1}{2}m_1(\dot{x_1}^2 + \dot{y_1}^2)$ を計算しても, 同じ結果になる.)

一方質点 m_2 について, そのデカルト座標 (x_2, y_2) から,

$$\begin{aligned}
&x_2 = l_1 \sin\varphi_1 + l_2 \sin\varphi_2 \quad , \quad y_2 = -(l_1 \cos\varphi_1 + l_2 \cos\varphi_2) \\
&\dot{x_2} = l_1\dot{\varphi_1}\cos\varphi_1 + l_2\dot{\varphi_2}\cos\varphi_2 \quad , \quad \dot{y_2} = l_1\dot{\varphi_1}\sin\varphi_1 + l_2\dot{\varphi_2}\sin\varphi_1 \\
&\dot{x}_2^2 = l_1^2\dot{\varphi_1}^2\cos^2\varphi_1 + 2l_1l_2\dot{\varphi_1}\dot{\varphi_2}\cos\varphi_1\cos\varphi_2 + l_2^2\dot{\varphi_2}^2\cos^2\varphi_2 \quad , \\
&\dot{y}_2^2 = l_1^2\dot{\varphi_1}^2\sin^2\varphi_1 + 2l_1l_2\dot{\varphi_1}\dot{\varphi_2}\sin\varphi_1\sin\varphi_2 + l_2^2\dot{\varphi_2}^2\sin^2\varphi_2
\end{aligned}$$

運動エネルギーとポテンシャル・エネルギーは,
$\cos(\alpha-\beta) = \cos\alpha\cos\beta + \sin\alpha\sin\beta$, また $\sin^2\alpha + \cos^2\alpha = 1$ だから

$$T_2 = \frac{1}{2}m_2(\dot{x_2}^2 + \dot{y_2}^2) = \frac{m_2}{2}\left(l_1^2\dot{\varphi_1}^2 + l_2^2\dot{\varphi_2}^2 + 2l_1l_2\cos(\varphi_1-\varphi_2)\dot{\varphi_1}\dot{\varphi_2}\right)$$

$$U_2 = -m_2gl_1\cos\varphi_1 - m_2gl_2\cos\varphi_2$$

従って,Lagrangian$L = T_1 + T_2 - U_1 - U_2$ は,

$$L = \frac{m_1+m_2}{2}l_1^2\dot{\varphi_1}^2 + \frac{m_2}{2}l_2^2\dot{\varphi_2}^2 + m_2l_1l_2\dot{\varphi_1}\dot{\varphi_2}\cos(\varphi_1-\varphi_2)$$

$$+ (m_1+m_2)gl_1\cos\varphi_1 + m_2gl_2\cos\varphi_2$$

になる.

$$\frac{\partial L}{\partial\varphi_1} = -m_2l_1l_2\sin(\varphi_1-\varphi_2)\dot{\varphi_1}\dot{\varphi_2} - (m_1+m_2)gl_1\sin\varphi_1 \quad ,$$

$$\frac{\partial L}{\partial\dot{\varphi_1}} = (m_1+m_2)l_1^2\dot{\varphi_1} + m_2l_1l_2\cos(\varphi_1-\varphi_2)\dot{\varphi_2} \quad ,$$

$$\frac{\partial L}{\partial\varphi_2} = m_2l_1l_2\sin(\varphi_1-\varphi_2)\dot{\varphi_1}\dot{\varphi_2} - m_2gl_2\sin\varphi_2 \quad ,$$

$$\frac{\partial L}{\partial\dot{\varphi_2}} = m_2l_2^2\dot{\varphi_2} + m_2l_1l_2\cos(\varphi_1-\varphi_2)\dot{\varphi_1}$$

より, Euler-Lagrange eq.

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial\dot{\varphi_i}} - \frac{\partial L}{\partial\varphi_i} = 0$$

は,

$$(m_1+m_2)l_1^2\ddot{\varphi_1} + m_2l_1l_2\cos(\varphi_1-\varphi_2)\ddot{\varphi_2} + m_2l_1l_2\sin(\varphi_1-\varphi_2)\dot{\varphi_1}\dot{\varphi_2} + (m_1+m_2)gl_1\sin\varphi_1 = 0,$$

$$m_2l_2^2\ddot{\varphi_2} + m_2l_1l_2\cos(\varphi_1-\varphi_2)\ddot{\varphi_1} - m_2l_1l_2\sin(\varphi_1-\varphi_2)\dot{\varphi_1}\dot{\varphi_2} + m_2gl_2\sin\varphi_2 = 0$$

2.2　Python による模擬実験

連立の常微分方程式にして, 関数 ode を定義する.

$$\frac{\mathrm{d}\varphi_1}{\mathrm{d}t} = \dot{\varphi}_1$$

$$\frac{\mathrm{d}\varphi_2}{\mathrm{d}t} = \dot{\varphi}_2$$

$$\frac{\mathrm{d}\dot{\varphi}_1}{\mathrm{d}t} = \frac{Mg\sin\varphi_1 + m_2\sin H(l_1\cos H + l_2)\dot{\varphi}_1\dot{\varphi}_2 - m_2 g\cos H\sin\varphi_2}{-(Ml_1 - m_2 l_1\cos^2 H)}$$

$$\frac{\mathrm{d}\dot{\varphi}_2}{\mathrm{d}t} = \frac{Mg\cos H\sin\varphi_1 + \sin H(Ml_1 + m_2 l_2\cos H)\dot{\varphi}_1\dot{\varphi}_2 - Mg\sin\varphi_2}{Ml_2 - m_2 l_2\cos^2 H}$$

ここで,　$H = \varphi_1 - \varphi_2$,　$M = m_1 + m_2$ である.

単振子との違いは質点が 2 個あるので, 関数 ode の戻り値は, 単振子のときに $[\varphi, \dot{\varphi}]$ としていたのを,$[\varphi_1, \dot{\varphi}_1, \varphi_2, \dot{\varphi}_2]$ とする.

積分は,scipy.integrate の odeint を, アニメーションは,matplotlib.animation の FuncAnimation を使う.

ソースコード 2.1　2 重平面振子

```
from numpy import sin, cos, pi
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
import numpy as np
from scipy.integrate import odeint
#
# Constants
#
G = 9.8              # [m/s^2]  acceleration of gravity
PHI1_0 = pi          # [rad] initial theta
PHI2_0 = -pi/6.      # [rad] initial theta
V1_0 = 0.            # [m/s]    initial velocity
V2_0 = 0.            # [m/s]    initial velocity
L1 = 1.              # [m]      length of pendulum
L2 = 1.              # [m]      length of pendulum
M1 = 1.              # [kg]     mass
M2 = 0.4             # [kg]     mass
DURATION = 15.       # [s]      duration time
INTERVAL = 0.05      # [s]      interval time
#
# Differential Equation
#
def ode(f, t):
    phi1, dphi1, phi2, dphi2 = f
    M, H = M1 + M2, phi1 - phi2
    dphi1dt = (M*G*sin(phi1)\
                + M2*(L2+L1*cos(H))*sin(H)*dphi1*dphi2\
```

```
                 - M2*G*cos(H)*sin(phi2))\
                 /(-M*L1+M2*L1*cos(H)*cos(H))
    dphi2dt = (M*G*cos(H)*sin(phi1)
                 +(M2*L2*cos(H)+M*L1)*sin(H)*dphi1*dphi2\
                 - M*G*sin(phi2))\
                 /(M*L2-M2*L2*(cos(H))**2)
    return [dphi1, dphi1dt, dphi2, dphi2dt]
#
# Initial Condition
#
f_0 = [PHI1_0, V1_0/L1, PHI2_0, V2_0/L2]
t = np.arange(0, DURATION + INTERVAL, INTERVAL)
#
# Solve the Equation
#
sol = odeint(ode, f_0, t)
phi1, phi2 = sol[:, 0], sol[:, 2]
x1 = L1 * sin(phi1)
y1 = - L1 * cos(phi1)
x2 = x1 + L2 * sin(phi2)
y2 = y1 - L2 * cos(phi2)
#
# Prepare the Screen to display
#
fig = plt.figure()
ax = fig.add_subplot(111, aspect='equal', autoscale_on=False,
                     xlim = [-L1 - L2, L1 + L2], ylim = [-L1 - L2, L1 + L2])
ax.grid()
markers_on = [1,2]
line, = plt.plot([], [], 'ro-', markevery=markers_on, animated = True)
time_template = 'time = %.1fs'
time_text = ax.text(0.05, 0.9, '', transform=ax.transAxes)
#
# Simulate
#
def init():
    time_text.set_text('')
    return line, time_text

def update(i):
    next_x = [0, x1[i], x2[i]]
    next_y = [0, y1[i], y2[i]]
    line.set_data(next_x, next_y)
    time_text.set_text(time_template % (i * INTERVAL))
    return line, time_text

FRAME_INTERVAL = 1000 * INTERVAL # [msec] interval time between frames
ani = FuncAnimation(fig, update, frames=np.arange(0, len(t)),
                    interval=FRAME_INTERVAL, init_func=init, blit=True)
#
# Show and Save the results
#
plt.show()
FPS = 1000/FRAME_INTERVAL        # frames per second
#ani.save('double_pendulum.mp4', fps=FPS, extra_args=['-vcodec', 'libx264'])
ani.save('double_pendulum.gif', writer='imagemagick', fps=FPS)
```

第 3 章

支点が水平方向に可動な単振子

3.1　モデルの定式化

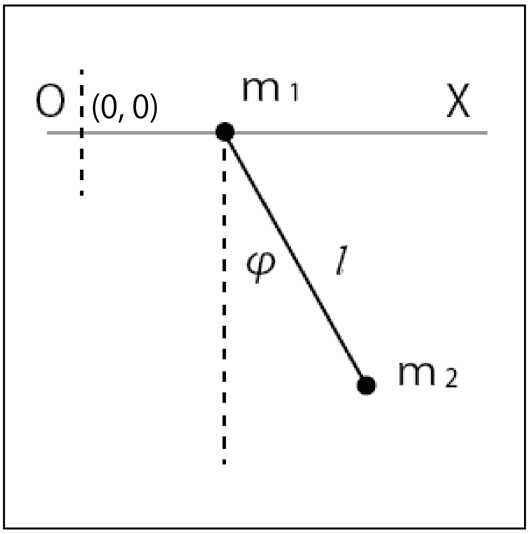

図 3.1　質量 m_1 の支点に質量 m_2 の単振子

質量 m_2 の単振子. その支点である質量 m_1 の質点が水平方向に運動できる.
水平に運動している質点 m_1 の運動エネルギーは、

$$T_1 = \frac{m_1}{2}\dot{x}^2$$

質点 m_2 のデカルト座標 (x_2, y_2) は、

$$\begin{aligned}
x_2 &= x + l\sin\varphi \quad , \quad y_2 = -l\cos\varphi \\
\dot{x}_2 &= \dot{x} + l\dot{\varphi}\cos\varphi \quad , \quad \dot{y}_2 = l\dot{\varphi}\sin\varphi \\
\dot{x}_2^2 &= \dot{x}^2 + 2\dot{x}l\dot{\varphi}\cos\varphi + l^2\dot{\varphi}^2\cos^2\varphi \quad , \quad \dot{y}_2^2 = l^2\dot{\varphi}^2\sin^2\varphi
\end{aligned}$$

質点 m_2 の運動エネルギーは、$\sin^2\alpha + \cos^2\alpha = 1$ を思い出して、

$$T_2 = \frac{m_2}{2}\left(\dot{x}_2^2 + \dot{y}_2^2\right) = \frac{m_2}{2}\left(\dot{x}^2 + 2l\dot{x}\dot{\varphi}\cos\varphi + l^2\dot{\varphi}^2\right)$$

一方、ポテンシャル・エネルギー U は、

$$U = -m_2 gl\cos\varphi$$

従って、Lagrangian$L = T_1 + T_2 - U$ は、

$$L = \frac{m_1+m_2}{2}\dot{x}^2 + \frac{m_2}{2}\left(l^2\dot{\varphi}^2 + 2l\dot{x}\dot{\varphi}\cos\varphi\right) + m_2 gl\cos\varphi$$

Euler-Lagrange eq. は, 次の計算をして、

$$\frac{\partial L}{\partial\dot{\varphi}} = m_2 l^2\dot{\varphi} + m_2 l\dot{x}\cos\varphi \qquad , \qquad \frac{\partial L}{\partial\varphi} = -m_2 l\dot{x}\dot{\varphi}\sin\varphi - m_2 gl\sin\varphi$$

$$\frac{\partial L}{\partial\dot{x}} = (m_1+m_2)\dot{x} + m_2 l\dot{\varphi}\cos\varphi \qquad , \qquad \frac{\partial L}{\partial x} = 0$$

従って,

$$m_2 l^2\ddot{\varphi} - m_2 l\dot{x}\sin\varphi = -m_2 l\dot{x}\dot{\varphi}\sin\varphi - m_2 gl\sin\varphi$$

$$\therefore \qquad \ddot{\varphi} = \frac{\sin\varphi}{l}\dot{x} + \frac{-\sin\varphi}{l}\dot{x}\dot{\varphi} + \frac{-g}{l}\sin\varphi$$

もう一方の式は,

$$(m_1+m_2)\ddot{x} + m_2 l\dot{\varphi}\cos\varphi = 0$$

$$\therefore \qquad \ddot{x} = \frac{-m_2 l\dot{\varphi}\cos\varphi}{m_1+m_2}$$

3.2　Python による模擬実験

これらを連立の一階微分方程式に直すと

$$\frac{\mathrm{d}\varphi}{\mathrm{d}t} = \dot{\varphi}$$

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \dot{x}$$

$$\frac{\mathrm{d}\dot{\varphi}}{\mathrm{d}t} = \frac{\sin\varphi}{l}\dot{x} + \frac{-\sin\varphi}{l}\dot{x}\dot{\varphi} + \frac{-g}{l}\sin\varphi$$

$$\frac{\mathrm{d}\dot{x}}{\mathrm{d}t} = \frac{-m_2 l\dot{\varphi}\cos\varphi}{m_1+m_2}$$

2 つの質点の座標 $(x, y), (x_2, y_2)$ については、座標 x はラグランジアンに陽に含まれていないことから循環的な座標である.(ランダウ「力学」p.37) 循環座標が存在する場合の運動方程式の積分は簡単化できて、x を含まない L を x で微分しても、その結果は定数 (ゼロ) になる.

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x} = 0$$

時間微分 $\frac{d}{dt}$ の結果がゼロなので、$\frac{\partial L}{\partial \dot{x}}$ は定数.

$$\frac{\partial L}{\partial \dot{x}} = (m_1 + m_2)\dot{x} + m_2 l \dot{\varphi}\cos\varphi = const.$$

これは系の運動量が保存することを表している. この式の積分は、$(m_1 + m_2)x + m_2 l \sin\varphi = const.$,$const = 0$ に対して x が次の通り求まる.（ランダウ「力学」p.42 問 3）

$$\begin{aligned} x &= \frac{-m_2 l \sin\varphi}{m_1 + m_2} \\ y &= 0 \\ x_2 &= x + l\sin\varphi \\ y_2 &= -y - l\cos\varphi \end{aligned}$$

支点の質量 m_1 を大きくして模擬実験を試すと、固定支点の状態に近づいてくる.($m_1 \to \infty$ で $x \to 0$, これは通常の単振子だ.)

ソースコード 3.1　支点が水平方向に可動な単振子

```
from numpy import sin, cos, pi
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
import numpy as np
from scipy.integrate import odeint
#
# Constants
#
L = 1                  # [m]       length of the pendulum
PHI0 = -pi/6           # [rad]     initial phi
V0 = 0.0               # [m/s]     initial velocity
X0 = 0                 # [m]       initial location
G  = 9.8               # [m/s^2]   acceleration of gravity
M1 =  1                # [kg]      mass
M2 =  1                # [kg]      mass
DURATION = 10          # [s]       duration time
INTERVAL = 0.05        # [s]       interval time
#
# Differential equation
#
def ode(f, t):
    phi, dphi, x, dx = f
    dpdt = sin(phi)/L*dx-sin(phi)/L*dx*dphi-G/L*sin(phi)
    dxdt = -M2*L*dphi*cos(phi)/(M1+M2)
    return [dphi, dpdt, dx, dxdt]
#
# Initial condition
#
f_0 = [PHI0, V0/L, X0, V0/L]
t = np.arange(0, DURATION + INTERVAL, INTERVAL)
#
# Solve the equation
```

```
#
sol = odeint(ode, f_0, t)
phi, x = sol[:, 0], sol[:, 2]
x1 = M2*L*sin(phi)/(M1+M2)
y1 = 0*t
x2 = x1 + L * sin(phi)
y2 = -y1 - L * cos(phi)
#
# Prepare the screen to display the results
#
fig = plt.figure()
ax = fig.add_subplot(111, aspect='equal', autoscale_on=False,\
                     xlim = [-L - L, L + L], ylim = [-L - L, L + L])
ax.grid()
ax.set_title('M1={}, M2={}, L={}'.format(M1,M2,L))
line, = plt.plot([], [], 'ro-', animated = True)
time_template = 'time = %.1fs'
time_text = ax.text(0.05, 0.9, '', transform=ax.transAxes)
#
# Simulate with animation
#
def init():
    time_text.set_text('')
    return line, time_text

def update(i):
    next_x = [x1[i], x2[i]]
    next_y = [y1[i], y2[i]]
    line.set_data(next_x, next_y)
    time_text.set_text(time_template % (i * INTERVAL))
    return line, time_text

FRAME_INTERVAL = 1000 * INTERVAL # [msec] interval time between frames
ani = FuncAnimation(fig, update, frames=np.arange(0, len(t)),
                    interval=FRAME_INTERVAL, init_func=init, blit=True)
#
# Show and save the results
#
FPS = 1000/FRAME_INTERVAL        # frames per second
plt.show()
#ani.save('pendulum3.mp4', fps=FPS, extra_args=['-vcodec', 'libx264'])
ani.save('pendulum3.gif', writer='imagemagick', fps=FPS)
```

第 4 章

支点が円周上を周回している単振子

4.1　モデルの定式化

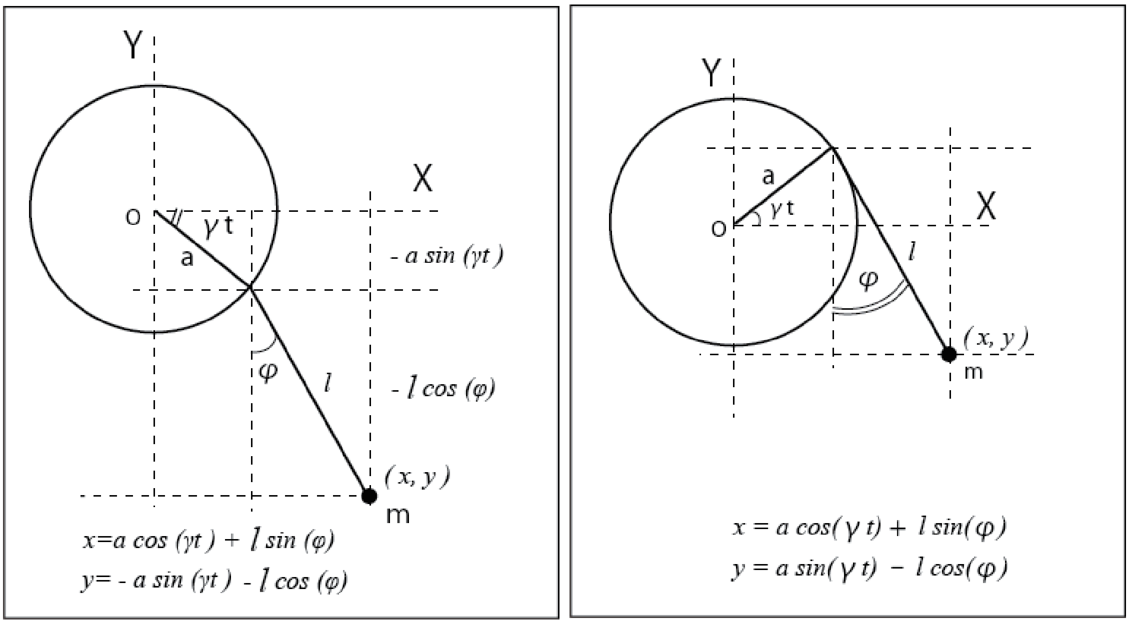

図 4.1　支点が円周上を周回する単振子

支点が鉛直平面内の円周上を一定の角速度 γ で一様に動いている単振子.（$\gamma < 0$ で CW,$\gamma > 0$ で CCW,γ が 0 なら支点は動かない)

円周上の点に注目して（$sin(-\gamma t) \quad = -sin(\gamma t)$ だから）CW と CCW の違い

右の図は,CCW 方向に回転する場合のモデル化になっている. 質点 m の位置 (x, y) は,

$$x = a\cos\gamma t + l\sin\varphi$$
$$y = a\sin\gamma t - l\cos\varphi$$

質点の速度, その自乗を作って,

$$\begin{aligned}
\dot{x} &= -a\gamma \sin \gamma t + l\dot{\varphi} \cos \varphi \\
\dot{y} &= a\gamma \cos \gamma t + l\dot{\varphi} \sin \varphi \\
\dot{x}^2 &= a^2\gamma^2 \sin^2 \gamma t + l^2\dot{\varphi}^2 \cos^2 \varphi - 2al\gamma\dot{\varphi} \sin \gamma t \cos \varphi \\
\dot{y}^2 &= a^2\gamma^2 \cos^2 \gamma t + l^2\dot{\varphi}^2 \sin^2 \varphi + 2al\gamma\dot{\varphi} \sin \varphi \cos \gamma t
\end{aligned}$$

運動エネルギー T とポテンシャル・エネルギー $U = mgy$ は,

$$\begin{aligned}
T &= \frac{m}{2}\left(\dot{x}^2 + \dot{y}^2\right) \\
&= \frac{m}{2}\{a^2\gamma^2 + l^2\dot{\varphi}^2 + 2al\gamma\dot{\varphi}\left(\sin \varphi \cos \gamma t - \cos \varphi \sin \gamma t\right)\} \\
&= \frac{m}{2}\{a^2\gamma^2 + l^2\dot{\varphi}^2 + 2al\gamma\dot{\varphi} \sin(\varphi - \gamma t)\} \\
U &= mgy = mga \sin \gamma t - mgl \cos \varphi
\end{aligned}$$

Lagrangian$L = T - U$ は,

$$\begin{aligned}
L &= T - U \\
&= \frac{m}{2}a^2\gamma^2 + \frac{m}{2}l^2\dot{\varphi}^2 + alm\gamma\dot{\varphi} \sin(\varphi - \gamma t) - mga \sin \gamma t + mgl \cos \varphi \\
&= \frac{m}{2}l^2\dot{\varphi}^2 + alm\gamma\dot{\varphi} \sin(\varphi - \gamma t) + mgl \cos \varphi + \left(\frac{m}{2}a^2\gamma^2 - mga \sin \gamma t\right)
\end{aligned}$$

右辺の最終項は, 次の関数　$f(t)$ の時間に関する完全導関数になっている.

$$f(t) = \frac{m}{2}a^2\gamma^2 t + mga\frac{1}{\gamma} \cos \gamma t, \quad \frac{df(t)}{dt} = \frac{m}{2}a^2\gamma^2 - mga \sin \gamma t$$

時間だけに依存する最終項を除けば,Lagrangian は最終的に次の様になる.(＊座標と時間の任意の関数 f の, 時間に関する完全導関数 $\dot{f}$ を Lagrangian の中に含んでいても, 作用積分によってその変分は消えてしまう項になるので, 取り除いても運動方程式としては変わらない.$\rightarrow$ ランダウ「力学」p.5)

$$\therefore L = \frac{m}{2}l^2\dot{\varphi}^2 + alm\gamma\dot{\varphi} \sin(\varphi - \gamma t) + mgl \cos \varphi$$

次の計算をして,

$$\begin{aligned}
\frac{\partial L}{\partial \dot{\varphi}} &= ml^2\dot{\varphi} + alm\gamma \sin(\varphi - \gamma t) \\
\frac{\partial L}{\partial \varphi} &= alm\gamma\dot{\varphi} \cos(\varphi - \gamma t) - mgl \sin \varphi
\end{aligned}$$

もう少し準備して,

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\varphi}} = ml^2\ddot{\varphi} - alm\gamma^2 \cos(\varphi - \gamma t)$$

Eular-Langange eq.

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\varphi}} - \frac{\partial L}{\partial \varphi} = 0$$

は,

$$\therefore \ddot{\varphi} = \frac{a\gamma^2}{l}\cos(\varphi - \gamma t) + \frac{a\gamma}{l}\dot{\varphi}\cos(\varphi - \gamma t) - \frac{g}{l}\sin\varphi$$

一方, 左の図は CW 方向に回転する場合のモデル化になっている.(実際, 右の図のモデルで γ を $-\gamma$ で置き換えても同じ事になる.) 質点 m の位置 (x, y) は,

$$\begin{aligned} x &= a\cos\gamma t + l\sin\varphi \\ y &= -a\sin\gamma t - l\cos\varphi \end{aligned}$$

質点の速度, その自乗と計算していくと,

$$\begin{aligned} \dot{x} &= -a\gamma\sin\gamma t + l\dot{\varphi}\cos\varphi \\ \dot{y} &= -a\gamma\cos\gamma t + l\dot{\varphi}\sin\varphi \\ \dot{x}^2 &= a^2\gamma^2\sin^2\gamma t + l^2\dot{\varphi}^2\cos^2\varphi - 2al\gamma\dot{\varphi}\sin\gamma t\cos\varphi \\ \dot{y}^2 &= a^2\gamma^2\cos^2\gamma t + l^2\dot{\varphi}^2\sin^2\varphi - 2al\gamma\dot{\varphi}\sin\varphi\cos\gamma t \end{aligned}$$

運動エネルギー T とポテンシャル・エネルギー U は,

$$\begin{aligned} T &= \frac{m}{2}\left(\dot{x}^2 + \dot{y}^2\right) \\ &= \frac{m}{2}\{a^2\gamma^2 + l^2\dot{\varphi}^2 - 2al\gamma\dot{\varphi}\left(\sin\varphi\cos\gamma t + \cos\varphi\sin\gamma t\right)\} \\ &= \frac{m}{2}\{a^2\gamma^2 + l^2\dot{\varphi}^2 - 2al\gamma\dot{\varphi}\sin(\varphi + \gamma t)\} \\ U &= mgy = -mga\sin\gamma t - mgl\cos\varphi \end{aligned}$$

Lagrangian$L = T - U$ は,

$$\begin{aligned} L &= T - U \\ &= \frac{m}{2}a^2\gamma^2 + \frac{m}{2}l^2\dot{\varphi}^2 - alm\gamma\dot{\varphi}\sin(\varphi + \gamma t) + mga\sin\gamma t + mgl\cos\varphi \\ &= \frac{m}{2}l^2\dot{\varphi}^2 - alm\gamma\dot{\varphi}\sin(\varphi + \gamma t) + mgl\cos\varphi + \left(\frac{m}{2}a^2\gamma^2 + mga\sin\gamma t\right) \end{aligned}$$

右辺の最終項は, 次の関数　$f(t)$ の時間に関する完全導関数になっている.

$$f(t) = \frac{m}{2}a^2\gamma^2 t - mga\frac{1}{\gamma}\cos\gamma t, \quad \frac{df(t)}{dt} = \frac{m}{2}a^2\gamma^2 + mga\sin\gamma t$$

(＊座標と時間の任意の関数 f の, 時間に関する完全導関数 $\dot{f}$ を Lagrangian の中に含んでいても, 作用積分によってその変分は消えてしまう項になるので, 取り除いても運動方程式としては変わらない.→ ランダウ「力学」p.5) 時間だけに依存する最終項を除けば,Lagrangian は最終的に次の様になる.

$$\therefore L = \frac{m}{2}l^2\dot{\varphi}^2 - alm\gamma\dot{\varphi}\sin(\varphi + \gamma t) + mgl\cos\varphi$$

次の計算をして,

$$\frac{\partial L}{\partial \dot{\varphi}} = ml^2\dot{\varphi} - alm\gamma\sin(\varphi+\gamma t)$$
$$\frac{\partial L}{\partial \varphi} = -alm\gamma\dot{\varphi}\cos(\varphi+\gamma t) - mgl\sin\varphi$$

もう少し準備して,

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\varphi}} = ml^2\ddot{\varphi} - alm\gamma^2\cos(\varphi+\gamma t)$$

Eular-Langange eq.

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\varphi}} - \frac{\partial L}{\partial \varphi} = 0$$

は,

$$\therefore \ddot{\varphi} = \frac{a\gamma^2}{l}\cos(\varphi+\gamma t) - \frac{a\gamma}{l}\dot{\varphi}\cos(\varphi+\gamma t) - \frac{g}{l}\sin\varphi$$

この CW の式は, CCW の場合の Eular-Lgrange eq. の γ を $-\gamma$ で置き換えて得られる式と同じ格好をしている.

4.2 Python による模擬実験

連立の一階微分方程式に直して, 関数 ode 定義する. 右の図（CCW モデル）の場合の Eular-Lagrange eq. からは,

$$\frac{\mathrm{d}\varphi}{\mathrm{d}t} = \dot{\varphi}$$

$$\frac{\mathrm{d}\dot{\varphi}}{\mathrm{d}t} = \frac{a\gamma^2}{l}\cos(\varphi-\gamma t) + \frac{a\gamma}{l}\dot{\varphi}\cos(\varphi-\gamma t) - \frac{g}{l}\sin\varphi$$

円周上の点の位置 (x_0, y_0) は、

$$x_0 = a\cos\gamma t \quad , \quad y_0 = a\sin\gamma t$$

であり、質点の位置 (x, y) は、

$$x = x_0 + l\sin\varphi \qquad , \quad y = y_0 - l\cos\varphi$$

である.

左の図（CW モデル）の場合の Eular-Lagrange eq. からは,

$$\frac{\mathrm{d}\varphi}{\mathrm{d}t} = \dot{\varphi}$$

$$\frac{\mathrm{d}\dot{\varphi}}{\mathrm{d}t} = \frac{a\gamma^2}{l}\cos(\varphi+\gamma t) - \frac{a\gamma}{l}\dot{\varphi}\cos(\varphi+\gamma t) - \frac{g}{l}\sin\varphi$$

円周上の点の位置 (x_0, y_0) は、

$$x_0 = a\cos\gamma t \quad , \quad y_0 = -a\sin\gamma t$$

であり、質点の位置 (x, y) は、

$$x = x_0 + l\sin\varphi \qquad , \quad y = y_0 - l\cos\varphi$$

である.

模擬実験によって, CW と CCW の動きを確認することができる. わざわざ場合分けしてまでの記述をしているが, （当然の事ながら,） 一方の定式化だけにしておいて、定数 GAMMA の符号を反転させても同じ現象になるよね.

ソースコード 4.1　支点が円周上を周回する単振子

```
import matplotlib.pyplot as plt
import numpy as np
from numpy import sin, cos, pi
from matplotlib.animation import FuncAnimation
from scipy.integrate import odeint
#
# Constants
#
CCW = False         # CW = not CCW
A = 0.8             # [m]        radius
F = 0.1             # [Hz]       frequency
GAMMA = 2.0*pi*F    # [rad/s]    angular velocity
G   = 9.8           # [m/s^2]    acceleration of gravity
PHI0 = pi/6.        # [rad]      initial angle
V0 = 0.05           # [m/s]      initial velocity
L = 1.              # [m]        length of the pendulum
DURATION = 20.      # [s]        duration time
INTERVAL = 0.05     # [s]        interval time
#
# Differential equation
#
def ode(f, t):
    phi, dphi = f
    if CCW:
        # CCW at the right figure
        dfdt = [dphi, (A*GAMMA*GAMMA/L)*cos(phi-GAMMA*t)\
                          + (A*GAMMA/L)*cos(phi-GAMMA*t)*dphi\
                          - (G/L)*sin(phi)]
    else:
        # CW at the left figure
        dfdt = [dphi, (A*GAMMA*GAMMA/L)*cos(phi+GAMMA*t)\
                          - (A*GAMMA/L)*cos(phi+GAMMA*t)*dphi\
```

```
                            - (G/L)*sin(phi)]
    return dfdt
#
# Initial condition
#
f0 = [PHI0, V0/L]    # [theta, v] at t = 0
t = np.arange(0, DURATION + INTERVAL, INTERVAL)
#
# Solve the equation
#
sol = odeint(ode, f0, t)
phi = sol[:, 0]
if CCW:
    # CCW at the right sided figure
    circ_x = A * cos(GAMMA*t)
    circ_y = A * sin(GAMMA*t)
    x = circ_x + L * sin(phi)
    y = circ_y - L * cos(phi)
else:
    # CW at the left sided figure
    circ_x = A * cos(GAMMA*t)
    circ_y = - A * sin(GAMMA*t)
    x = circ_x + L * sin(phi)
    y = circ_y - L * cos(phi)
#
# Prepare the screen to display
#
fig = plt.figure()
ax = fig.add_subplot(111, aspect='equal', autoscale_on=False,\
                     xlim=(-2*L, 2*L), ylim=(-2*L, 2*L))
ax.grid()
markers_on = [2]
line, = plt.plot([], [], 'ro-', markevery=markers_on,  animated = True)
time_template = 'time = %.1fs'
time_text = ax.text(0.05, 0.9, '', transform=ax.transAxes)
#
# Simulate
#
def init():
    time_text.set_text('')
    return line, time_text

def update(i):
    next_x = [0, circ_x[i], x[i]]
    next_y = [0, circ_y[i], y[i]]
    line.set_data(next_x, next_y)
    time_text.set_text(time_template % (i*INTERVAL))
    return line, time_text

FRAME_INTERVAL = 1000 * INTERVAL # [msec] interval between frames
ani = FuncAnimation(fig, update, frames=np.arange(0, len(t)),
                    interval=FRAME_INTERVAL, init_func=init, blit=True)
#
# Show and save the results
#
plt.show()
FPS = 1000/FRAME_INTERVAL # frames per second
if CCW:
    fname = 'pendulum4CCW'
```

```
else:
    fname = 'pendulum4CW'
#ani.save(fname+'.mp4', fps=FPS, extra_args=['-vcodec', 'libx264'])
ani.save(fname+'.gif', writer='imagemagick', fps=FPS)
```

第 5 章

支点が鉛直に振動している単振子

5.1 モデルの定式化

支点が鉛直方向に $a\cos\gamma t$ にしたがって振動している単振子
単振子の座標は、$(l\sin\varphi, l\cos\varphi)$ なので、質点 m の座標は、$x = l\sin\varphi, y = a\cos\gamma t + l\cos\varphi$
次の準備をしておいて

$$\begin{aligned}
\dot{x} &= l\dot{\varphi}\cos\varphi \quad , \quad \dot{y} = -a\gamma\sin\gamma t - l\dot{\varphi}\sin\varphi \\
\dot{x}^2 &= l^2\dot{\varphi}^2\cos^2\varphi \quad , \quad \dot{y}^2 = a^2\gamma^2\sin^2\gamma t + 2a\gamma l\dot{\varphi}\sin\gamma t\sin\varphi + l^2\dot{\varphi}^2\sin^2\varphi
\end{aligned}$$

運動エネルギー T とポテンシャル・エネルギー U は次の通り（$\sin^2\alpha + \cos^2\alpha = 1$）

$$\begin{aligned}
T &= \frac{m}{2}\left(\dot{x}^2 + \dot{y}^2\right) \\
&= \frac{m}{2}\left(l^2\dot{\varphi}^2\left(\sin^2\varphi + \cos^2\varphi\right) + a^2\gamma^2\sin^2\gamma t + 2a\gamma l\dot{\varphi}\sin\gamma t\sin\varphi\right) \\
&= \frac{m}{2}l^2\dot{\varphi}^2 + ma\gamma l\dot{\varphi}\sin\gamma t\sin\varphi + \frac{m}{2}a^2\gamma^2\sin^2\gamma t \\
U &= -mgy \\
&= -mg(a\cos\gamma t + l\cos\varphi) \\
&= -mga\cos\gamma t - mgl\cos\varphi
\end{aligned}$$

ラグランジアン $L = T - U$ を求める.

$$\begin{aligned}
L &= T - U \\
&= \frac{m}{2}l^2\dot{\varphi}^2 + ma\gamma l\dot{\varphi}\sin\gamma t\sin\varphi + \frac{m}{2}a^2\gamma^2\sin^2\gamma t + mga\cos\gamma t + mgl\cos\varphi \\
&= \frac{m}{2}l^2\dot{\varphi}^2 + \left(ma\gamma^2 l\cos\gamma t\cos\varphi - ma\gamma l\frac{d}{dt}(\cos\varphi\sin\gamma t)\right) \\
&\qquad\qquad + \frac{m}{2}a^2\gamma^2\sin^2\gamma t + mga\cos\gamma t + mgl\cos\varphi \\
&= \frac{m}{2}l^2\dot{\varphi}^2 + ma\gamma^2 l\cos\gamma t\cos\varphi + mgl\cos\varphi \\
&\qquad + \left(-ma\gamma l\frac{d}{dt}(\cos\varphi\sin\gamma t) + \frac{m}{2}a^2\gamma^2\sin^2\gamma t + mga\cos\gamma t\right)
\end{aligned}$$

ここで、次の関係を使っている.

$$
\begin{aligned}
-ma\gamma l \frac{d}{dt}(\cos\varphi \sin\gamma t) &= -ma\gamma l\left(-\dot{\varphi}\sin\varphi\sin\gamma t + \gamma\cos\varphi\cos\gamma t\right) \\
&= ma\gamma l\dot{\varphi}\sin\gamma t\sin\varphi - ma\gamma^2 l\cos\gamma t\cos\varphi
\end{aligned}
$$

更に、次の関数 $f(\varphi, t)$ を用意すると、

$$
f(\varphi, t) = -ma\gamma l\cos\varphi\sin\gamma t + \frac{1}{4}ma^2\gamma^2 t - \frac{1}{8}ma^2\gamma\sin 2\gamma t + mga\frac{1}{\gamma}\sin\gamma t
$$

ラグランジアンの最後の括弧内の部分は、関数 $f(\varphi, t)$ の時間に関する完全導関数になっている.（$\cos 2\alpha = 1 - 2\sin^2\alpha$）

$$
\begin{aligned}
\frac{df(\varphi, t)}{dt} &= -ma\gamma l(-\dot{\varphi}\sin\varphi\sin\gamma t + \gamma\cos\varphi\cos\gamma t) + \frac{1}{4}ma^2\gamma^2 - \frac{1}{8}ma^2\gamma 2\gamma\cos 2\gamma t + mga\frac{1}{\gamma}\gamma\cos\gamma t \\
&= -ma\gamma l\frac{d}{dt}(\cos\varphi\sin\gamma t) + \frac{1}{4}ma^2\gamma^2 - \frac{1}{4}ma^2\gamma^2(1 - 2\sin^2\gamma t) + mga\cos\gamma t \\
&= -ma\gamma l\frac{d}{dt}(\cos\varphi\sin\gamma t) + \frac{1}{4}ma^2\gamma^2 2\sin^2\gamma t + mga\cos\gamma t \\
&= -ma\gamma l\frac{d}{dt}(\cos\varphi\sin\gamma t) + \frac{m}{2}a^2\gamma^2\sin^2\gamma t + mga\cos\gamma t
\end{aligned}
$$

従って、最終的な Lagrangian は、（時間に関する完全導関数の部分を除いて）

$$
L = \frac{ml^2}{2}\dot{\varphi}^2 + ma\gamma^2 l\cos\gamma t\cos\varphi + mgl\cos\varphi
$$

次の計算から

$$
\begin{aligned}
\frac{\partial L}{\partial\dot{\varphi}} &= ml^2\dot{\varphi} \\
\frac{\partial L}{\partial\varphi} &= -mla\gamma^2\cos\gamma t\sin\varphi - mgl\sin\varphi
\end{aligned}
$$

Euler-Lagrange eq. は

$$
\ddot{\varphi} = -\frac{a\gamma^2}{l}\cos\gamma t\sin\varphi - \frac{g}{l}\sin\varphi
$$

5.2　Python による模擬実験

これを連立の一階微分方程式に直すと

$$
\begin{aligned}
\frac{\mathrm{d}\varphi}{\mathrm{d}t} &= \dot{\varphi} \\
\frac{\mathrm{d}\dot{\varphi}}{\mathrm{d}t} &= -\frac{a\gamma^2}{l}\cos\gamma t\sin\varphi - \frac{g}{l}\sin\varphi
\end{aligned}
$$

質点の座標は

$$x = l \sin\varphi \qquad , \quad y = a\cos\gamma t + l\cos\varphi$$

ソースコード 5.1　支点が鉛直に振動している単振子

```
import matplotlib.pyplot as plt
import numpy as np
from numpy import sin, cos, pi
from matplotlib.animation import FuncAnimation
from scipy.integrate import odeint
#
# Constants
#
A = 1.0          # [m]
F = 0.1          # [Hz]
GAMMA = 2*pi*F
G  = 9.8         # [m/s^2] gravitational acceleration
THETA0 = pi/4    # [rad]   initial angle
V0 = 0.05        # [m/s]   initial velocity
L = 1            # [m]     length of the pendulum
DURATION = 20    # [s]     duration time
INTERVAL = 0.05  # [s]     interval time
#
# Differential equation
#
def ode(f, t):
    theta, dtheta = f
    dfdt = [dtheta, -1*(A*GAMMA*GAMMA/L) * cos(GAMMA * t) * sin(theta) - (G/L) *
      sin(theta)]
    return dfdt
#
# Initial condition
#
f0 = [THETA0, V0/L]    # [theta, v] at t = 0
t = np.arange(0, DURATION + INTERVAL, INTERVAL)
#
# Solve the equation
#
sol = odeint(ode, f0, t)
theta = sol[:, 0]
x = L * sin(theta)
y = - L * cos(theta) - A * cos(GAMMA*t)
#
# Prepare the screen to display the results
#
fig = plt.figure()
ax = fig.add_subplot(111, aspect='equal', autoscale_on=False, xlim=(-2*L, 2*L),
  ylim=(-2*L, 2*L))
ax.grid()
markers_on = [1]
line, = plt.plot([], [], 'r-o', markevery=markers_on, animated=True)
time_template = 'time = %.1fs'
time_text = ax.text(0.05, 0.9, '', transform=ax.transAxes)
#
# Simulate the results
#
def init():
```

```
    time_text.set_text('')
    return line, time_text

def update(i):
    circ_x = 0*t
    circ_y = -A * cos(GAMMA*t)
    next_x = [circ_x[i], x[i]]
    next_y = [circ_y[i], y[i]]
    line.set_data(next_x, next_y)
    time_text.set_text(time_template % (i*INTERVAL))
    return line, time_text

FRAME_INTERVAL = 1000 * INTERVAL # [msec] interval between frames

ani = FuncAnimation(fig, update, frames=np.arange(0, len(t)),
                    interval=FRAME_INTERVAL, init_func=init, blit=True)
#
# Show and save the results
#
plt.show()
FPS = 1000/FRAME_INTERVAL # frames per second
#ani.save('pendulum5.mp4', fps=FPS, extra_args=['-vcodec', 'libx264'])
# using the Pillow instead of the imagemagic for MovieWriter
ani.save('pendulum5.gif', writer='Pillow', fps=FPS)
```

第 6 章

支点が水平に振動している単振子

6.1　モデルの定式化

単振子の座標は、$(l\sin\varphi, l\cos\varphi)$ なので、質点 m の座標は、

$$x = a\cos\gamma t + l\sin\varphi, y = l\cos\varphi$$

予め次の計算の準備をしておいて、

$$\begin{aligned}\dot{x} &= -a\gamma\sin\gamma t + l\dot{\varphi}\cos\varphi \quad , \quad \dot{y} = -l\dot{\varphi}\sin\varphi \\ \dot{x}^2 &= a^2\gamma^2\sin^2\gamma t - 2a\gamma l\dot{\varphi}\sin\gamma t\cos\varphi + l^2\dot{\varphi}^2\cos^2\varphi \quad , \quad \dot{y}^2 = l^2\dot{\varphi}^2\sin^2\varphi\end{aligned}$$

運動エネルギー T とポテンシャル・エネルギー U は次のようになる（$\sin^2\alpha + \cos^2\alpha = 1$ を使って）

$$\begin{aligned}T &= \frac{m}{2}(\dot{x}^2 + \dot{y}^2) \\ &= \frac{m}{2}l^2\dot{\varphi}^2\left(\sin^2\varphi + \cos^2\varphi\right) - ma\gamma l\dot{\varphi}\sin\gamma t\cos\varphi + \frac{m}{2}a^2\gamma^2\sin^2\gamma t \\ &= \frac{m}{2}l^2\dot{\varphi}^2 - ma\gamma l\dot{\varphi}\sin\gamma t\cos\varphi + \frac{m}{2}a^2\gamma^2\sin^2\gamma t \\ U &= -mgy = -mgl\cos\varphi\end{aligned}$$

ラグランジアン $L = T - U$ を求めていく.

$$\begin{aligned}L &= T - U \\ &= \frac{m}{2}l^2\dot{\varphi}^2 - ma\gamma l\dot{\varphi}\sin\gamma t\cos\varphi + \frac{m}{2}a^2\gamma^2\sin^2\gamma t + mgl\cos\varphi \\ &= \frac{m}{2}l^2\dot{\varphi}^2 + \left(ma\gamma^2 l\cos\gamma t\sin\varphi - ma\gamma l\frac{d}{dt}(\sin\varphi\sin\gamma t)\right) + \\ &\qquad\qquad \frac{m}{2}a^2\gamma^2\sin^2\gamma t + mgl\cos\varphi \\ &= \frac{m}{2}l^2\dot{\varphi}^2 + ma\gamma^2 l\cos\gamma t\sin\varphi + \\ &\qquad \left(-ma\gamma l\frac{d}{dt}(\sin\varphi\sin\gamma t)\frac{m}{2}a^2\gamma^2\sin^2\gamma t\right) + mgl\cos\varphi\end{aligned}$$

ここで、次の関係を使っている.

$$
\begin{aligned}
-ma\gamma l \frac{d}{dt}(\sin\varphi\sin\gamma t) &= -ma\gamma l\left(\gamma\sin\varphi\cos\gamma t + \dot{\varphi}\cos\varphi\sin\gamma t\right) \\
&= -ma\gamma^2 l\sin\varphi\cos\gamma t - ma\gamma l\dot{\varphi}\cos\varphi\sin\gamma t
\end{aligned}
$$

更に、次の関数 $f(\varphi, t)$ を用意する、

$$
f(\varphi, t) = -ma\gamma l\sin\varphi\sin\gamma t + \frac{1}{4}ma^2\gamma^2 t - \frac{1}{8}ma^2\gamma\sin 2\gamma t
$$

ラグランジアンの最後の括弧内の部分は、関数 $f(\varphi, t)$ の時間に関する完全導関数になっている.（$\cos 2\alpha = 1 - 2\sin^2\alpha$ を使って）

$$
\begin{aligned}
\frac{df(\varphi, t)}{dt} &= -ma\gamma l\frac{d}{dt}(\sin\varphi\sin\gamma t) + \frac{1}{4}ma^2\gamma^2 - \frac{1}{8}ma^2\gamma(2\gamma)\cos 2\gamma t \\
&= -ma\gamma l\frac{d}{dt}(\sin\varphi\sin\gamma t) + \frac{1}{4}ma^2\gamma^2 - \frac{1}{4}ma^2\gamma^2(1 - 2\sin^2\gamma t) \\
&= -ma\gamma l\frac{d}{dt}(\sin\varphi\sin\gamma t) + \frac{1}{4}ma^2\gamma^2(2\sin^2\gamma t) \\
&= -ma\gamma l\frac{d}{dt}(\sin\varphi\sin\gamma t) + \frac{m}{2}a^2\gamma^2\sin^2\gamma t
\end{aligned}
$$

従って、最終的な Lagrangian は、（時間に関する完全導関数を除いて）

$$
L = \frac{ml^2}{2}\dot{\varphi}^2 + mla\gamma^2\cos\gamma t\sin\varphi + mgl\cos\varphi
$$

次の計算から,

$$
\begin{aligned}
\frac{\partial L}{\partial\dot{\varphi}} &= ml^2\dot{\varphi} \\
\frac{\partial L}{\partial\varphi} &= mla\gamma^2\cos\gamma t\cos\varphi - mgl\sin\varphi
\end{aligned}
$$

Euler-Lagrange eq. は

$$
\ddot{\varphi} = \frac{a\gamma^2}{l}\cos\gamma t\cos\varphi - \frac{g}{l}\sin\varphi
$$

6.2　Python による模擬実験

これを連立の一階微分方程式に直すと

$$
\begin{aligned}
\frac{\mathrm{d}\varphi}{\mathrm{d}t} &= \dot{\varphi} \\
\frac{\mathrm{d}\dot{\varphi}}{\mathrm{d}t} &= \frac{a\gamma^2}{l}\cos\gamma t\cos\varphi - \frac{g}{l}\sin\varphi
\end{aligned}
$$

質点の座標は

$$
x = a\cos\gamma t + l\sin\varphi \qquad , \quad y = l\cos\varphi
$$

ソースコード 6.1　支点が水平に振動している単振子

```
import matplotlib.pyplot as plt
import numpy as np
from numpy import sin, cos, pi
from matplotlib.animation import FuncAnimation
from scipy.integrate import odeint

#
# Constants
#
A = 1.0 # [m]
F = 0.1 # [Hz]
GAMMA = 2*pi*F
G   = 9.8           # [m/s^2] gravitational acceleration
THETA0 = pi/4       # [rad]   initial angle
V0 = 0.05              # [m/s]   initial velocity
L = 1               # [m]     length of the pendulum
DURATION = 20       # [s]     duration time
INTERVAL = 0.05     # [s]     interval time
#
# Differential equation
#
def ode(f, t):
    theta, dtheta = f
    dfdt = [dtheta, (A*GAMMA*GAMMA/L) * cos(GAMMA * t) * cos(theta) - (G/L) * sin(theta)]
    return dfdt
#
# Initial condition
#
f0 = [THETA0, V0/L]     # [theta, v] at t = 0
t = np.arange(0, DURATION + INTERVAL, INTERVAL)
#
# Solve the equation
#
sol = odeint(ode, f0, t)
theta = sol[:, 0]
x = L * sin(theta) + A * cos(GAMMA*t)
y = -L * cos(theta)
#
# Prepare the screen to display the results
#
fig = plt.figure()
ax = fig.add_subplot(111, aspect='equal', autoscale_on=False, xlim=(-2*L, 2*L), ylim=(-2*L, 2*L))
ax.grid()
markers_on = [1]
line, = plt.plot([], [], 'ro-', markevery=markers_on, animated = True)
time_template = 'time = %.1fs'
time_text = ax.text(0.05, 0.9, '', transform=ax.transAxes)
#
# Simulate with animation
#
def init():
    time_text.set_text('')
    return line, time_text

```

```
def update(i):
    circ_x = A * cos(GAMMA*t)
    circ_y = 0*t
    next_x = [circ_x[i], x[i]]
    next_y = [circ_y[i], y[i]]
    line.set_data(next_x, next_y)
    time_text.set_text(time_template % (i*INTERVAL))
    return line, time_text

FRAME_INTERVAL = 1000 * INTERVAL # [msec] interval between frames

ani = FuncAnimation(fig, update, frames=np.arange(0, len(t)),
                    interval=FRAME_INTERVAL, init_func=init, blit=True)
#
# Show and save the results
#
FPS = 1000/FRAME_INTERVAL # frames per second
plt.show()
#ani.save('pendulum4.mp4', fps=FPS, extra_args=['-vcodec', 'libx264'])
# using Pillow instead of imagemagic for MovieWriter
ani.save('pendulum4.gif', writer='pillow', fps=FPS)
```

謝辞

参考文献

[1]