

常 Ordinary Differential Equation

数値計算による微分方程式解法(python版)

cc by Shigeto R. Nishitani

- /Users/bob/Desktop/maple_ode/python_ode.ipynb
- origin git@github.com:daddygongon/maple_ode.git (fetch)

Table of Contents

- 1 Euler法による落下運動
 - 1.1 重力場中の運動
 - 1.2 Euler法
 - 1.3 重力場中の運動をEuler法で解いたら
 - 1.4 空気抵抗がある水滴の落下
- 2 高精度計算
 - 2.1 バネの運動
 - 2.2 2次のRunge-Kuttaの導出
 - 2.3 Runge-Kutta2次公式
 - 2.4 Runge-Kutta4次公式
 - 2.5 連立方程式にRunge-Kutta4次公式を
- 3 RLC回路の応答
- 4 課題
 - 4.1 雨粒
 - 4.2 大砲
- 5 自由課題
 - 5.1 RLC回路
 - 5.2 RLC回路

Euler法による落下運動 ¶

重力場中の運動

重力場中のボールの落下を考えて、1軸で考えた運動方程式を立てます。

mass gravity Force

$$v = \frac{dx}{dt}$$

$$a = \frac{d^2x}{dt^2}$$

質量を m , 重力加速度を g として、働く力が $F = -mg$ であるとする、ニュートンの運動方程式 $F = ma$ は、

$$-mg = m \frac{d^2x}{dt^2}$$

となります。

外力 ← | → 内力

Free gravitation fall

速度 velocity

The diagram shows a ball of mass m falling under gravity mg . The differential equation is $m \frac{dv}{dt} = -mg$. The velocity update is $v_{i+1} = v_i - g dt$. The graph shows position x vs time t with discrete steps $i, i+1$.

Euler法

1次の微分方程式の一般形は

$$\frac{dx}{dt} = f(x, t)$$

と書けます。この微分方程式を簡単な近似から求めるオイラー法を示します。 $x(t + \delta t)$ をテイラー級数展開すると、

$$x(t + \delta t) \approx x(t) + \frac{dx}{dt} \delta t$$

となります。これらを代入すると、計算アルゴリズムはつぎのようになります、

$$x_{i+1} = x_i + f_i \delta t$$

ここで、 f_i は点 (x_i, t_i) における関数の値です。このアルゴリズムを適用して、 $t + \delta t$ の座標 x_{i+1} を一つ前の時間の座標 x_i から導くことができます。これを重力場中の運動方程式に適用します。

重力場中の運動をEuler法で解いたら

Euler法は一階の微分方程式に対する定式化をしています。ところが、重力場中の運動は2階の微分方程式です。このようなときには媒介変数を導入して1次連立方程式に置き直します。

媒介変数として速度 v を使って、2階の運動方程式

$$\frac{d^2x}{dt^2} = -g$$

が、1階の連立方程式

$$\frac{dv}{dt} = -g$$

$$\frac{dx}{dt} = v$$

で置き換えられると考えることに相当します。アルゴリズムにすると、

$$v_{i+1} = v_i - g \, dt$$

$$x_{i+1} = x_i + v_i \, dt$$

なる連立方程式を解くことに置き換わります。これをpythonで関数にして、さらに計算結果を表示させてみます。

```
In [1]: def euler(x0, v0):
        v1 = v0 - g * dt
        x1 = x0 + v0 * dt
        return x1, v1
```

Eulerは x_i, v_i を受け取って、先ほど導いた簡単な計算によって、 v_{i+1}, x_{i+1} を順次計算して返します。結果は、

```
In [2]: import matplotlib.pyplot as plt
```

```
def my_plot(xx, vv, tt):
    plt.plot(tt, xx, color = 'b', linestyle='--', label="height")
    plt.plot(tt, vv, color = 'r', label="velocity")
    plt.legend()
    plt.xlabel('time')
    plt.ylabel('height and velocity')
    plt.grid()
    plt.show()
```

```
In [14]: g, dt = 9.8, 0.01
          # g, dt = 9.8, 0.1
          # tt, xx, vv = [0.0], [0.0], [9.8] #for compare rain_drop
          tt, xx, vv = [0.0], [0.0], [9.80]
```

```
t = 0.0

for i in range(0,220): #250 for compare rain_drop
    t += dt
    x, v = euler(xx[-1], vv[-1])
    tt.append(t)
    xx.append(x)
    vv.append(v)
```

```
my_plot(xx, vv, tt)
```

