数値計算論　後期中間レポート
常微分方程式の数値解法

人間情報システム工学科 4 年 36 号
松山　京介

2021 年 11 月 22 日

演習課題 3

課題 1

次の 1 階微分方程式をオイラー法で解け．ただし，$h = 0.1,\ 0 \leq x \leq 2.5$ とせよ．

$$\frac{dy}{dx} = xy, \quad \text{初期条件 } x = 0 \text{ で } y = 1$$

得られた結果を解析解 $y = e^{x^2/2}$ とともにグラフで表し比較せよ．

課題 1.1　アルゴリズム

テイラー展開の 1 次近似を元にした解法である．$y' = f(x, y)$（ここで $y' = \frac{d}{dx}y(x)$）の解 $y(x)$ に対して，テイラーの定理より：

$$y(x+h) = y(x) + hy'(x) + \frac{h^2}{2!}y''(x) + \cdots + \frac{h^p}{p!}y^{(p)}(x) + \cdots \tag{1}$$

ここで，h を刻み，

$$x = x_i,\ x_{i+1} = x_i + h,\ y_i = y(x_i),\ y_i' = f(x_i, y_i),\ y_{i+1} = y(x_i + h) \tag{2}$$

として点 (x_i, y_i) から微分方程式を満たす次の点 (x_{i+1}, y_{i+1}) は，(1) より，$p = 1$ として 1 次近似するとするとオイラー法の数値解法：

$$y_{i+1} = y_i + hy_i' = y_i + hf(x_i, y_i) \tag{3}$$

となる．この解法は微分を次の差分商で置き換えたものと考えられる．

$$\frac{y(x+h) - y(x)}{h} \approx f(x, y(x)) = \frac{dy}{dx} \tag{4}$$

よって，オイラー法は (1) より h^2 程度の誤差を含んでいる．

オイラー法の数値解法のアルゴリズムを図 1 に示す．

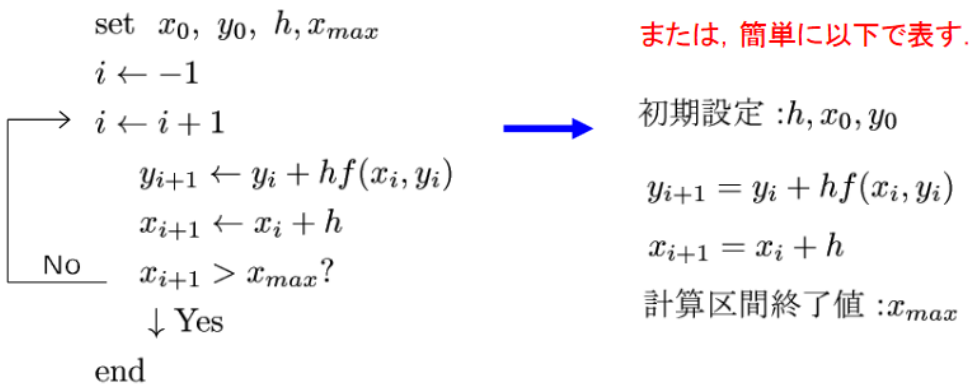

図 1: オイラー法のアルゴリズム

プログラムリスト

ソースコード 1: オイラー法のプログラム

```
import matplotlib.pyplot as plt
import math

h=0.1
x=[]
y=[]
y2=[]
x.append(0)
y.append(1)
i=0
x_m=2.5

def f(x,y):
    return x*y
while(1):
    y.append(y[i]+h*f(x[i],y[i]))
    x.append(x[i]+h)
    if(x[i+1]>x_m):
        break
    i+=1

for i in x:
    y2.append(math.exp(i*i/2))

plt.plot(x,y,label="Euler")
plt.plot(x,y2,label="analysis")
plt.legend(loc=0)
plt.show()
```

実行結果

課題１の実行結果を以下に示す

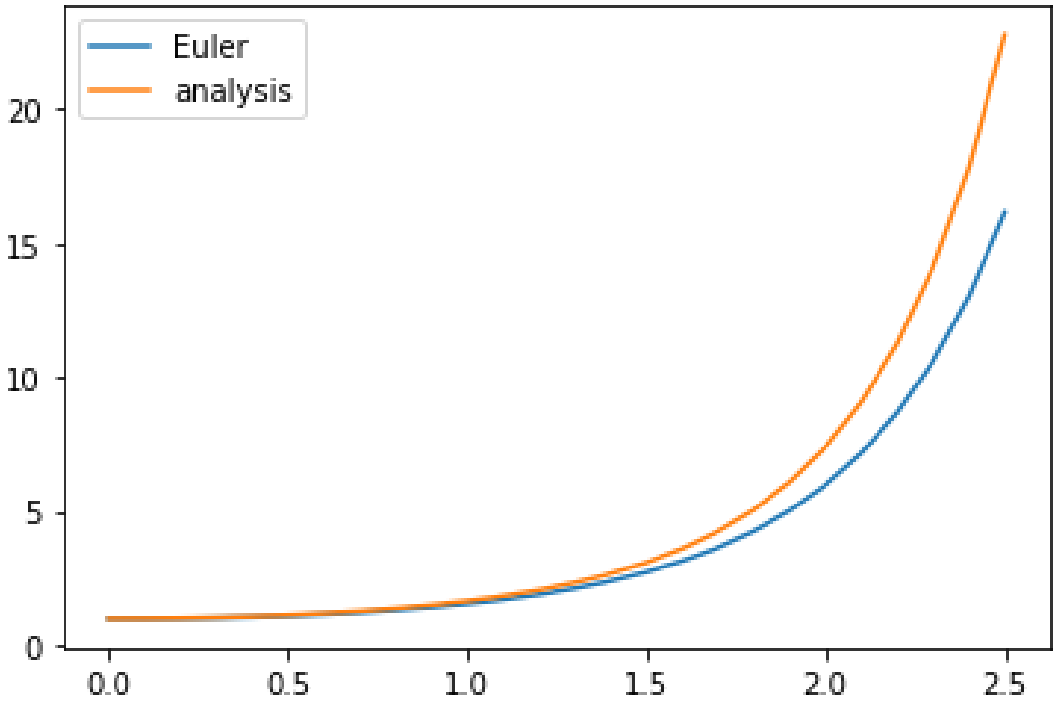

図 2

考察

問題の１階微分方程式を数学的に解いた式を下に示す

$$\frac{dy}{dx} = xy$$
$$\frac{1}{y}dy = xdx$$
$$\int \frac{1}{y}dy = \int xdx$$
$$log|y| = \frac{1}{2}x^2 + C(C : 積分定数)$$
$$y = e^{\frac{x^2}{2}} \cdots 解析解$$

オイラー法では微分を差分商で置き換えたものと考え近似させるため誤差は h^2 程度になる．この問題では h = 0.1 としているので理論上の誤差は 0.01 程度である．図 2 を見ると関数の傾きが緩やかな所では近似率が高く傾きが急な所では近似率が比較的低い

課題 2

次の 2 階微分方程式をオイラー法で解け．ただし，$h = 0.4,\ 0 \leq x \leq 6$ とせよ．

$$\frac{d^2y}{dx^2} = e^x - y - \frac{dy}{dx},\ \text{初期条件 } x = 0 \text{ で } y = 1,\ \frac{dy}{dx} = 1$$

得られれた結果を解析解：

$$y = \frac{2}{3}e^{-\frac{x}{2}}\left(\cos\frac{\sqrt{3}}{2}x + \sqrt{3}\sin\frac{\sqrt{3}}{2}x\right) + \frac{e^x}{3}$$

とともにグラフで表し比較せよ．

課題 2.1　プログラムリスト

ソースコード 2: オイラー法のプログラム

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3  import math
4
5  h = 0.4
6  x0 = 0
7  y0 = 1
8  yb_0 = 1
9
10 x1 = [x0]
11 y1 = [y0]
12 yb_1 = [yb_0]
13
14 i = -1
15
16 x = np.arange(0, 6+h, h)
17 y = ( 2 * (np.exp(-x/2)) * (np.cos(x*np.sqrt(3)/2) + np.sqrt(3)*np.sin(x*np.sqrt
       (3)/2)) + np.exp(x))/3
18
19 def fx(x,y):
20     return pow(math.e,x) - y - yb_0
21
22 while x1[i+1] < 6:
23     i    += 1
24     y1  += [ y1[i] + fx(x1[i], y1[i])*h]
25     x1  += [ x1[i] + h ]
26
27 plt.plot(x,y,label="Euler")
```

```
plt.plot(x1,y1,label="analysis")
plt.legend(bbox_to_anchor=(0, 1), loc='upper left', borderaxespad=0, fontsize=9)
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

課題 2.2　実行結果

課題 2 の実行結果を以下に示す.

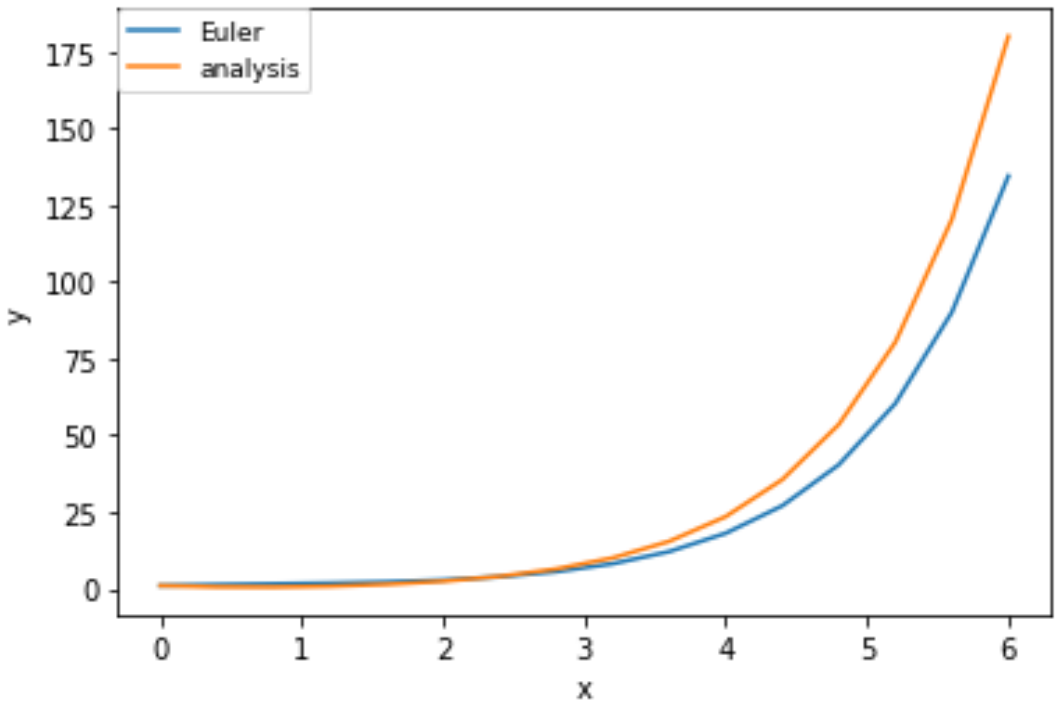

図 3

課題 2.3　考察

二階微分方程式におけるオイラー法の解き方として一階微分方程式の形に置き換える方法を用いた.

$y = f(x), \dfrac{dy}{dx} = z(x)$ とおくと

$$\frac{df(x)}{dx} = z(x)$$

$$\frac{d^2y}{dx^2} = \frac{d^2f(x)}{dx^2} = \frac{dz(x)}{dx}$$

$$\frac{d^2y}{dx^2} = \frac{dz(z)}{dx}$$

問題文$\dfrac{d^2y}{dx^2} = e^x - y - \dfrac{dy}{dx}$を解くと

$$\frac{dz(z)}{dx} = e^x - f(x) - z(x)$$

このように一階微分方程式の形にすることできた. あとは, 図 1 のアルゴリズム通りに書くと実装が可能になる. オイラー法のデメリットとして傾きの大きいところでの近似率が悪いというところが挙げられる.

課題 3

次の微分方程式の境界値問題を解け．ただし，分割数 $n = 10$ とせよ．

$$\frac{d^2x}{dx^2} - 4y = 0, \quad \text{境界条件：} x = 0 \text{ で } y = 0,\ x = 1 \text{ で } y = 0.25 \tag{5}$$

得られた結果を解析解：

$$y = \frac{1}{4}\frac{e^{2x} - e^{-2x}}{e^2 - e^{-2}}$$

とともにグラフで表し比較せよ．

課題 3.1　アルゴリズム

2 点以上の点 x に対する y または y の微分値が得られている問題を境界値問題という．次の 2 階微分方程式について考える．

$$\begin{aligned} &\frac{d^2}{dx^2}y + A(x)\frac{d}{dx}y + B(x)y = C(x) \\ &\text{境界条件 :} x = a \text{ のとき } y = c,\ x = b \text{ のとき } y = d. \end{aligned} \tag{6}$$

このとき境界条件は図 4 のようになる．この解法は差分方程式に置き換えて，連立 1 次方程式を解く問題に帰着させる．

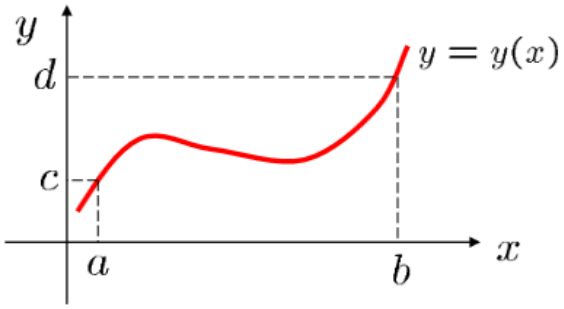

図 4: 境界条件

まず，$y_i = y(x_i)$，$y_{i+1} = y(x_i + h)$, h:刻み幅とする．テイラー展開より，

$$y_{i+1} = y_i + hy_i' + h^2\frac{y_i''}{2!} + h^3\frac{y_i^{(3)}}{3!} + \cdots \tag{7}$$

また，(7) の h を $-h$ とすると，

$$y_{i-1} = y_i - hy_i' + h^2\frac{y_i''}{2!} - h^3\frac{y_i^{(3)}}{3!} + \cdots \tag{8}$$

h^3 以上の項は無視できるほど小さいものと仮定すると，(7)−(8) から

$$y_i' = \frac{1}{2h}(y_{i+1} - y_{i-1}) \tag{9}$$

同様にして (7)+(8) から h^3 以上の項を無視すると，

$$y_i'' = \frac{1}{h^2}(y_{i+1} - 2y_i + y_{i-1}) \tag{10}$$

(9) と (10) の近似値を (6) に代入すると，次の差分方程式が得られる．

$$\frac{1}{h^2}(y_{i+1} - 2y_i + y_{i-1}) + A(x_i)\frac{1}{2h}(y_{i+1} - y_{i-1}) + B(x_i)y_i = C(x_i) \tag{11}$$

この (11) は $i = 1, 2, \ldots, n-1$ について成立するので未知数 $y_1, y_2, \ldots, y_{n-1}$ は，対角要素付近以外はすべて 0 となるので扱いやすい．

課題 3.2　プログラムリスト

ソースコード 3: 境界値のプログラム

```
1 import matplotlib.pyplot as plt
2 import math
3 import numpy as np
4
5 n=10
6 h=(1-0)/n
7
8 x_1=0
```

```
y_1=0
x_2=1
y_2=10

M=[]

N=[
   [-(2+4*h**2),1,0,0,0,0,0,0,0],
   [1,-(2+4*h**2),1,0,0,0,0,0,0],
   [0,1,-(2+4*h**2),1,0,0,0,0,0],
   [0,0,1,-(2+4*h**2),1,0,0,0,0],
   [0,0,0,1,-(2+4*h**2),1,0,0,0],
   [0,0,0,0,1,-(2+4*h**2),1,0,0],
   [0,0,0,0,0,1,-(2+4*h**2),1,0],
   [0,0,0,0,0,0,1,-(2+4*h**2),1],
   [0,0,0,0,0,0,0,1,-(2+4*h**2)]
   ]

N_inv=np.linalg.inv(N)

for i in range(1,n):
    if(i==n-1):
        M+=[-y_2]
    else:
        M+=[0]

L=np.dot(N_inv,M).tolist()

L.insert(0,y_1)
L.append(y_2)

x=np.arange(0,1+h,h)
y=(np.e**(2*x)-np.e**(-2*x))/(4*(np.e**(2)-np.e**(-2)))

plt.plot(x,y,label="boundary")
plt.legend(loc=0)
plt.show()
```

課題 3.3　実行結果

課題 3 の実行結果を以下にに示す.

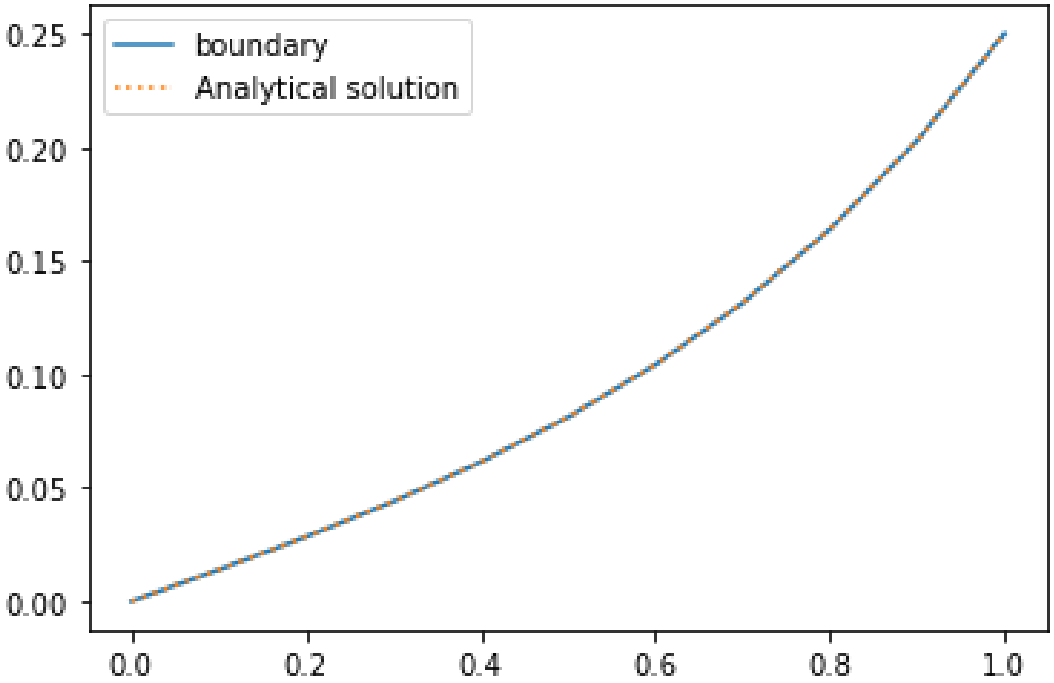

図 5

課題 3.4　考察

図 5 より，誤差がほぼ生じていないないことがわかる．今回の問題では分割数 n を 10 に設定し実装したが誤差も少なく近似率が高いことが見受けられた．オイラー法と比べ誤差も少なくコンパクトに実装できるが始点と終点が分からないと実装できず終点がわからないグラフ等を近似する際には向かない．

課題 4

次の 1 階微分方程式をオイラー法，2 次のルンゲクッタ法，3 次のルンゲクッタ法と 4 次のルンゲクッタ法で解け．ただし，$h = 0.5,\ 1 \leq x \leq 20$ とせよ．

$$\frac{dy}{dx} = \frac{y}{2x}, \quad \text{初期条件 } x = 1 \text{ で } y = 1$$

得られた結果を解析解 $y = \sqrt{x}$ とともにグラフで表し比較せよ．

課題 4.1　アルゴリズム

テイラー展開の 2 次近似を元にした解法が 2 次のルンゲクッタ法である．以下に 2 次のルンゲクッタ法の導出を行う．

$$y_{i+1} = y_i + ahf\left(x_i, y_i\right) + bhf\left(x_i + \ell h, y_i + \ell hf\left(x_i, y_i\right)\right) \tag{12}$$

と y_{i+1} が点 (x_i, y_i) での式で近似できるものと役定する．ただし，a, b, ℓ は定数．ここで，右辺第 3 項はテイラー展開より

$$\begin{aligned} bhf\left(x_i + \ell h, y_i + \ell h f\left(x_i, y_i\right)\right) &= bh\left\{f\left(x_i, y_i\right) + \ell h \frac{d}{dx} f\left(x_i, y_i\right) + O\left(h^2\right)\right\} \\ &= bh\left\{f\left(x_i, y_i\right) + \ell h f_x\left(x_i, y_i\right) + \ell h \cdot \frac{\partial y}{\partial x} \cdot \frac{\partial}{\partial y} f\left(x_i, y_i\right) + O\left(h^2\right)\right\} \\ &= bh\left\{f\left(x_i, y_i\right) + \ell h f_x\left(x_i, y_i\right) + \ell h f\left(x_i, y_i\right) \cdot f_y\left(x_i, y_i\right) + O\left(h^2\right)\right\} \end{aligned}$$

ただし，

$$f_x\left(x_i, y_i\right) = \left.\frac{\partial f(x, y)}{\partial x}\right|_{x=x_i, y=y_i}, f_y\left(x_i, y_i\right) = \left.\frac{\partial f(x, y)}{\partial y}\right|_{x=x_i, y=y_i}$$

$O\left(h^2\right)$：テーラー展開の 2 次以降の項．よって (12) 式は，

$$y_{i+1} = y_i + h(a+b) f\left(x_i, y_i\right) + b\ell h^2\left\{f_x\left(x_i, y_i\right) + f\left(x_i, y_i\right) \cdot f_y\left(x_i, y_i\right)\right\} + O\left(h^3\right) \tag{13}$$

ただし，$O\left(h^3\right)$：テーラー展開の 3 次以降の項．

また，関数 $y(x)$ の点 x_{i+1} における値 y_{i+1} はテイラー展開より

$$\begin{aligned} y_{i+1} &= y_i + h y_i' + \frac{h^2}{2} y_i'' + O\left(h^3\right) \\ &= y_i + h f\left(x_i, y_i\right) + \frac{h^2}{2}\left\{f_x\left(x_i, y_i\right) + f_y\left(x_i, y_i\right) f\left(x_i, y_i\right)\right\} + O\left(h^3\right) \end{aligned} \tag{14}$$

(13) 式 (14) 式となるには，$a = 0, b = 1, \ell = \frac{1}{2}$ として (12) 式より次の 2 次のルンゲクッタ の近似式が得られる．

$$y_{i+1} \approx y_i + h f\left(x_i + \frac{h}{2}, y_i + \frac{1}{2} h f\left(x_i, y_i\right)\right)$$

常微分方程式：$y' = f(x, y)$ の解を求める 2 次のルンゲクッタ法のアルゴリ ズム: 初期条件：h, x_0, y_0

$$\begin{gathered} y_{i+1} = y_i + h f\left(x_i + \frac{h}{2}, y_i + \frac{k_1}{2}\right) = y_i + k_2 \\ \text{ただし，} k_1 = h f\left(x_i, y_i\right), k_2 = h f\left(x_i + \frac{h}{2}, y_i + \frac{k_1}{2}\right) \end{gathered}$$

2 次のルンゲクッタ法の誤差は (13),(14) 式より h^3 程度である．

同様にして 3 次，4 次のルンゲクッタ法が定義できる．以下はそのアルゴリズムである．

3次のルンゲクッタ法（誤差はh^4 程度）

(Runge-Kutta 3rd Order Method)：

初期条件 $:h, x_0, y_0$

$$y_{i+1} = y_i + \frac{k_1 + 4k_2 + k_3}{6} \tag{1.11}$$

ただし，

$$k_1 = hf(x_i, y_i),$$

$$k_2 = hf(x_i + \frac{h}{2}, y_i + \frac{k_1}{2})$$

$$k_3 = hf(x_i + h, y_i + 2k_2 - k_1)$$

4次のルンゲクッタ法（誤差はh^5 程度）

(Runge-Kutta 4th Order Method)：

初期条件 $:h, x_0, y_0$

$$y_{i+1} = y_i + \frac{k_1 + 2k_2 + 2k_3 + k_4}{6}$$

ただし，

$$k_1 = hf(x_i, y_i),$$

$$k_2 = hf(x_i + \frac{h}{2}, y_i + \frac{k_1}{2})$$

$$k_3 = hf(x_i + \frac{h}{2}, y_i + \frac{k_2}{2})$$

$$k_4 = hf(x_i + h, y_i + k_3)$$

課題 4.2　プログラムリスト

```
import matplotlib.pyplot as plt
import math
import numpy as np

h = 0.5
x_m = 20
x_euler = []
y = []
y_euler = []
y_runge2 = []
y_runge3 = []
y_runge4 = []
x_euler.append (1)
y_euler.append (1)
y_runge4.append (1)
y_runge3.append (1)
y_runge2.append (1)
n=2

def f(x):
    return math.pow(x,n)

def f_euler (x,y):
    return (2*y)/x

i = 0
while (1):
    y_euler.append(y_euler[i]+h*f_euler(x_euler[i],y_euler[i]))
    x_euler . append ( x_euler [ i] + h )
    if( x_euler [i +1] > x_m ):
        break
    i += 1

for i in x_euler :
    y. append (f(i))

x_tmp = 1

for i in range ( len ( x_euler ) -1):
    y_runge2 . append ( y_runge2 [ i] + h * f_euler ( x_tmp , y_runge2 [i ]))
    x_tmp += h

x_tmp = 1
```

```
for i in range ( len ( x_euler ) -1):
    k1 = h* f_euler ( x_tmp , y_runge3 [i])
    k2 = h* f_euler ( x_tmp + h /2 , y_runge3 [i]+ k1/2)
    k3 = h* f_euler ( x_tmp +h , y_runge3 [i]+2*k2-k1 )

    y_runge3 . append ( y_runge3 [i] +( k1 +4* k2 + k3 )/6)
    x_tmp += h
x_tmp = 1

for i in range (len(x_euler)-1):
    k1=h*f_euler(x_tmp,y_runge4[i])
    k2=h*f_euler(x_tmp + h /2 , y_runge4 [i ]+ k1 /2)
    k3=h*f_euler(x_tmp + h /2 , y_runge4 [i ]+ k2 /2)
    k4=h*f_euler(x_tmp+h,y_runge4[i]+k3)

    y_runge4.append(y_runge4[i]+(k1+2*k2+2*k3+k4)/6)
    x_tmp += h

plt.plot(x_euler,y,label= " Analysis ")
plt.plot(x_euler,y_runge2,label = "Runge2nd")
plt.plot(x_euler,y_runge3,label = "Runge3rd")
plt.plot(x_euler,y_runge4,label = "Runge4th")
plt.plot(x_euler,y_euler,label = " Eular ")
plt.legend(loc=0) # 凡例
plt.show()
```

課題 4.3　実行結果

課題４の実行結果を以下にに示す.

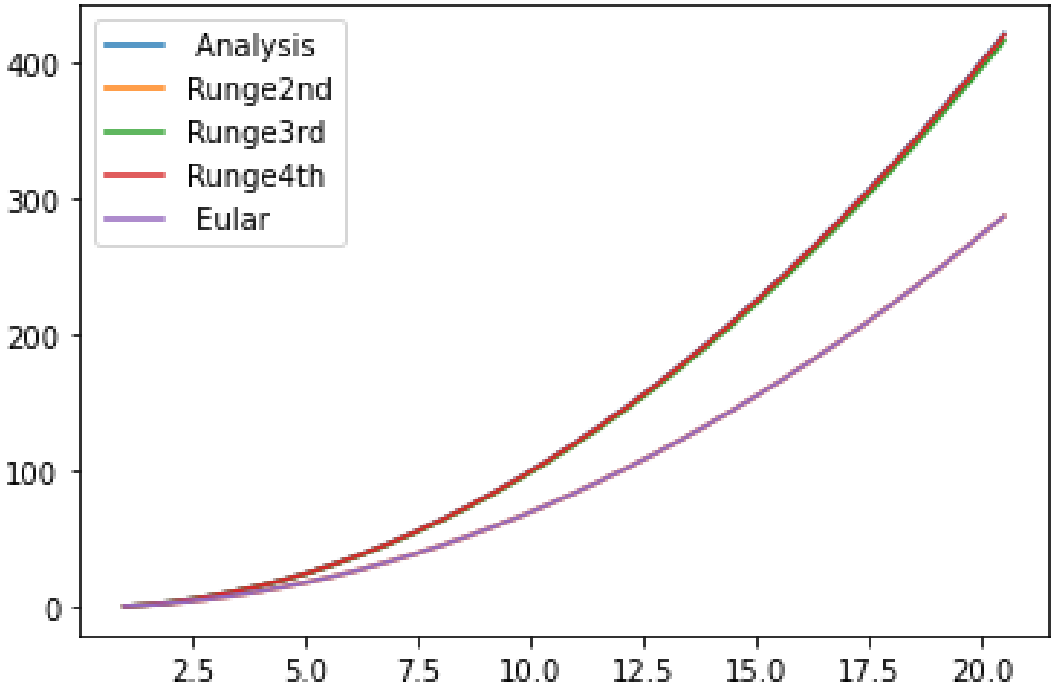

図 6

課題 4.4　考察

オイラー法の解よりもルンゲクッタ法を用いた解の方が解析解に近くなっていくことが確認できる．ルンゲクッタ法ではテイラー展開の n 次近似をもとに作られている．例えば 2 次のルンゲクッタ法ではテイラー展開の二次近似を元にするためテイラー展開の 3 次以降の項は $O(h^3)$ とし無視している．この時 2 次の誤差は h^3 程度である．

感想

解析解の導出に様々な方法を用いて近似できることが確認できたが, 非常に骨の折れる作業だった．ルンゲクッタ法に関してはかなりの精度で近似できるが, オイラー法は近似の精度が低いことが確認できた．