3 C プログラムを用いた微分方程式実習

3.1 ファイルのダウンロードと展開

次の手順に従って，プログラム２つをダウンロードして，自分のディレクトリに準備する．

演習室での授業のはじめに行う

1. ターミナルソフトウェアで，この授業専用のディレクトリを作成する．

```
cd ~
mkdir DE
cd DE
```

2. Firefox など，WWW ブラウザを起動し，本授業の web ページを開く．
あるいは以下の URL を指定する．
`https://www.oit.ac.jp/labs/is/system/shinkai/lecture/`
さらに「微分方程式」を開き，`DE1st.c` と `DE2nd.c` のファイルを２つダウンロードする.（それぞれファイル名を右クリックして「名前をつけててリンク先を保存」する）．
ダウンロードしたファイルは，先ほど作成した DE ディレクトリに入れる．

3. ターミナルで，`ls` して，２つのファイル（`DE1st.c` と `DE2nd.c`）があることを確認しよう．この２つのプログラムファイルは，何度も書き換えるので，不用意に破壊しないように，はじめにコピーをとっておくとよい．

```
cp DE1st.c DE1st_original.c
cp DE2nd.c DE2nd_original.c
```

3.2 基本的な利用方法

いくつかの課題に対して，次のことを行う．

1. プログラムファイル（`DE1st.c`, `DE2nd.c`）を必要に応じて編集する．

`DE1st.c`	１階の微分方程式を Euler 法で解くプログラム．解析解もプロットできる．
`DE2nd.c`	２階の微分方程式を Euler 法で解くプログラム．解析解もプロットできる．

2. プログラムをコンパイルする．`DE1st.c` のプログラムをコンパイルするときには，

```
gcc -o DE1st.exe DE1st.c -lm
```

`-lm` は，math.h をインクルードするためのオプションである．`-o` の直後は生成される実行ファイルの名前になる．

3. プログラムを実行する．
上記のコマンドを用いてコンパイルすると，実行ファイルは，`DE1st.exe` になるので，

```
./DE1st.exe
```

4. プログラムは，２つのファイル（`output.numerical`, `output.analytic`）を結果として生成する．

`output.numerical`	Euler 法で解いた結果ファイル．式の入力，初期条件の設定が正しければ，それなりの正しい結果になるはず．
`output.analytic`	解析解ファイル．自分で解いた答えを関数として入力しておき，その数値を出力する．自分の解と入力が正しければ，数値解と一致するはず．

両者を gnuplot でグラフにして，一致しているかどうかを確かめる．gnuplot を開き，

```
gnuplot
gnuplot> plot "output.numerical", "output.analytic"
```

図を点ではなく，線で描くときには

```
gnuplot> plot "output.numerical" with line, "output.analytic" with line
```

あるいは

```
gnuplot> plot "output.numerical" w l, "output.analytic" w l
```

gnuplot を終了するときは

```
gnuplot> quit
```

3.3 DE1st.c

プログラムファイル DE1st.c である．解読せよ．

どこを書き換えたら良いか，を理解すること．

```c
// Solve 1st order differential equation using Euler method
// C
// compile as ::   gcc -lm -o DE1st.exe DE1st.c
// execute as ::   ./DE1st.exe
// output files :: output.analytic     t  x
//                 output.numerical    t  x
#include <stdio.h>
#include <math.h>
//
#define X0 2.0        /* Initial Value  x(0)   初期値  */
#define T0 0.0        /* starting time t0 */
#define TMAX 10.0     /* ending time tmax */
#define OUTPUTSTEP 10 /* output data every xx steps */
// ------------------------------------------------------------
// differential equation
//     dxdt = right-hand side of the 1st order DE
double rhs(double x, double t){
    double dxdt ;
    dxdt =  -0.5 * x + exp(-0.2 * t) ; // 例を書いている．一番最後の行の式が実行される
    dxdt =  -0.5 * x + sin(t) ;
    dxdt =  -0.2 * x * t;
    dxdt =  -0.5 * x ;
    return dxdt;
}
// ------------------------------------------------------------
// analytic solution
double sol(double x, double t){
    double solution;
    solution = exp(t); // 例を書いている．一番最後の行の式が実行される
    solution = 2.0 * exp(-0.5 * t);
    return solution;
}
// ------------------------------------------------------------
int main(void)
{
  char filename1[] = "output.numerical";
  char filename2[] = "output.analytic";
  FILE *fp1, *fp2;
  double dt=0.01; // delta t
  double t,x;
  int icount=0;
  // open files
  fp1 = fopen(filename1, "w");
  fp2 = fopen(filename2, "w");
  // initial set up
  t = T0;
  x = X0;
    printf("dt= %8.4f \n",dt);
    printf("     t        numerical      analytic      diff \n");
  // t-loop
  while(t < TMAX){
    // check accuracy and output
      if(icount % OUTPUTSTEP == 0){
          // output
          printf("%10.3f %11.5f %11.5f %12.8f \n", t,x,sol(x,t),x-sol(x,t));
          fprintf(fp1,"%12.5f %12.5f\n", t,x);
          fprintf(fp2,"%12.5f %12.5f\n", t,sol(x,t));
      }
      icount += 1;
    // Forward Difference
    x += dt * rhs(x,t);
    // next t
    t += dt;
  } // end of t-loop
  // close files
  fclose(fp1);
  fclose(fp2);
}
```

4　Python で書いたプログラム（参考）

Python にはグラフ化するツールも備わっている．ここでは，2 階の微分方程式

$$\frac{d^2x}{dt^2} = -4x$$

を解くプログラムを例として挙げる．まずは，オイラー法で計算した結果をリスト t_plot[]，x_plot[]，a_plot[] に格納する．

```
# Solve 2nd order differential equation using Euler method
# Python 3 version
import math
import numpy as np
#  equation of motion
def dvdt(t,x):
    return -4.0*x
#  analytic solution
def sol(t,x0):
    return x0 * math.cos (2.0 * t)
# set parameters
t0  =0.0        # starting time
tmax=10.0       # ending time
dt  =0.01       # delta t
nend=int(tmax/dt)+1
outputstep = 10 # output data every xx steps
# data for plotting graph
i=0
iend=int(tmax/dt/outputstep)+1
t_plot = np.zeros(iend)
x_plot = np.zeros(iend)
a_plot = np.zeros(iend)
#
x0=1.0   # initial value x(0)
v0=0.0   # initial value v(0)
# initial set up
t=t0
x=x0
dxdt=v0
t_plot[i] = t
x_plot[i] = x
a_plot[i] = sol(t,x0)
# first line of the data
print('{:^12}{:^12}{:^12}{:^12}'.format('i   t',
                           'numerical','analytic','diff'))
print(f'{i:4d}{t:6.2f}{x:12.5f}{sol(t,x0):12.5f}{x-sol(t,x0):12.5f}')
# t-loop
for n in range(1, nend):
#   forward difference
    dxdt = dxdt + dt * dvdt(t,x)
    x    = x    + dt * dxdt
#   next t
    t = t + dt
    if np.mod(n,outputstep) == 0:
        i=i+1
        t_plot[i] = t
        x_plot[i] = x
        a_plot[i] = sol(t,x0)
        print(f'{i:4d}{t:6.2f}{x:12.5f}{sol(t,x0):12.5f}{x-sol(t,x0):12.5f}')
```

上記のプログラムで得られた t_plot[]，x_plot[]，a_plot[] をグラフにする．

```
import matplotlib.pyplot as plt
fig=plt.subplots(figsize=(8,6))
plt.plot(t_plot, x_plot,'.', c='k',label='numerical',markersize=6)
plt.plot(t_plot, a_plot, c='r',label='analytic')
plt.xlabel('t')
plt.ylabel('x')
plt.grid()
plt.legend(loc='lower right')
```

4.1 微分方程式の計算【1 階の微分方程式】

$\boxed{\text{C.1}}$ プログラム `DE1st.c` と `DE2nd.c` で用いているのは，微分方程式を解く手段としては，最も基本的な前進 Euler 法と呼ばれるものである．教科書を読んで，原理を理解せよ．

教科書 §7.1.3 と §7.1.4.

$\boxed{\text{C.2}}$ 以下の問題を解き，その答えを得た後，プログラム `DE1st.c` で問題となる微分方程式と解答となる解析解を入力し，両者が一致することを確かめよ．

(1) $y' = -2y, \quad y(x=0) = 2$
(2) $y' = 3y, \quad y(x=0) = 0.1$
(3) $y' = y(y-2), \quad y(x=0) = 1$
(4) $y' = y(y-2), \quad y(x=0) = -1$
(5) $y\,y' + x = 0, \quad y(x=0) = 2$
(6) $y' + y = 2x^2 + 4x - 1, \quad y(x=0) = 2$
(7) $y' + y = 2e^x, \quad y(x=0) = 1$
(8) $y' - y = 2e^{-x}, \quad y(x=0) = 0$
(9) $y' + 2y = e^{-2x}, \quad y(x=0) = -2$
(10) $y' + 3y = 5\sin x - 5\cos x, \quad y(x=0) = 2$

$\boxed{\text{C.3}}$ 積分の部分を，前進 Euler 法ではなく，台形公式やシンプソン公式を用いて改良してみよう．

余裕のある人のみ．

4.2 微分方程式の計算【2 階の微分方程式】

$\boxed{\text{C.4}}$ プログラム `DE2nd.c` では，2 階微分方程式を解いているが，どのように解いているか，解読せよ．

$\boxed{\text{C.5}}$ 関数 $y(t)$ について以下の微分方程式を解け．プログラム `DE2nd.c` で問題となる微分方程式と解答となる解析解を入力し，両者が一致することを確かめよ．

(1) $y'' + 4y = 0, \quad y(t=0) = 2, \ y'(t=0) = 0$
(2) $y'' + 4y = 0, \quad y(t=0) = 0, \ y'(t=0) = 2$
(3) $y'' - 4y = 0, \quad y(t=0) = 1, \ y'(t=0) = 0$
(4) $y'' - 4y = 0, \quad y(t=0) = 1, \ y'(t=0) = -1$

$\boxed{\text{C.6}}$ 関数 $y(t)$ について以下の微分方程式を解け．プログラム `DE2nd.c` で問題となる微分方程式と解答となる解析解を入力し，両者が一致することを確かめよ．

(6),(7) は $x = [0, 30]$ で plot せよ．

(1) $y'' - y' - 6y = 0, \quad y(t=0) = 1, \ y'(t=0) = -1$
(2) $y'' - y' - 6y = 0, \quad y(t=0) = 1, \ y'(t=0) = -2$
(3) $y'' + 4y' + 4y = 0, \quad y(t=0) = 2, \ y'(t=0) = 6$
(4) $y'' + 2y' + 10y = 0, \quad y(t=0) = 3, \ y'(t=0) = 0$
(5) $y'' + 2y' - 8y = 18e^{-t}, \quad y(t=0) = -2, \ y'(t=0) = 1$
(6) $y'' + 5y' + 6y = 5\sin t, \quad y(t=0) = 3, \ y'(t=0) = 0$
(7) $y'' + 4y = \sin 2t, \quad y(t=0) = 1, \ y'(t=0) = 0$

$\boxed{\text{C.7}}$ Runge-Kutta 法を用いて積分できるように，プログラムを改良せよ．

教科書 §7.1.5 参照．余裕のある人のみ．