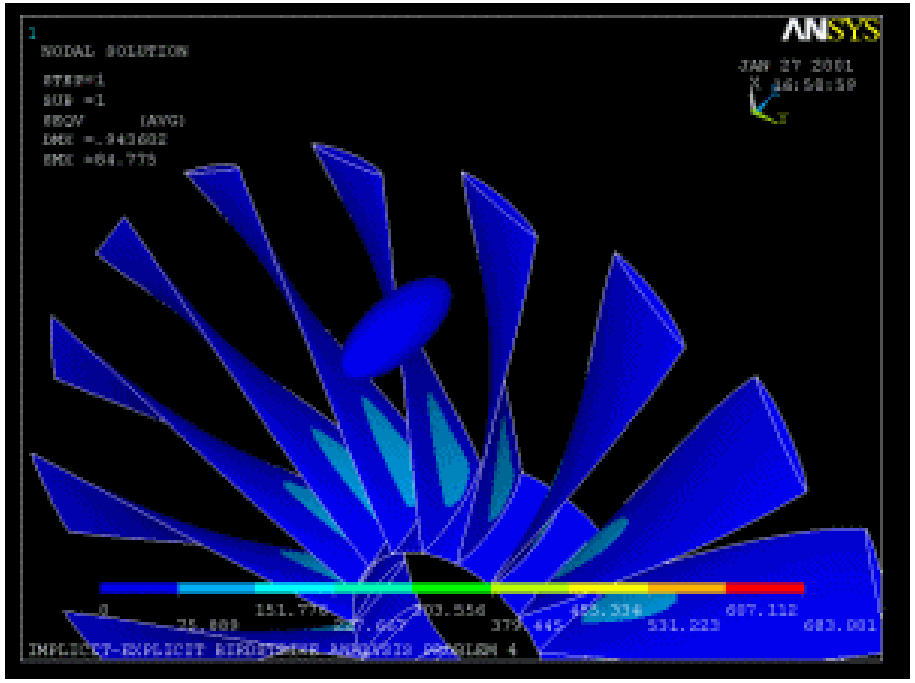
数値解析

2023年度前期　第13週 [7月13日]



静岡大学大学院
工学研究科機械工学専攻
ロボット・計測情報分野
創造科学技術大学院
情報科学専攻

三浦 憲二郎

講義アウトライン [7月13日]

- 常微分方程式
  - 1段階法と多段階法
  - オイラー法
  - ホイン法

微分方程式の重要性

- 自然現象や工学モデルの記述
  - 微分方程式が用いられる.
- 常微分方程式と偏微分方程式

理科系の数学入門 (斎藤秀司・三松佳彦・戸瀬信之編)
第4巻 常微分方程式 高崎金久著, 日本評論社

常微分方程式の起源は物理学, 特に力学の研究にある. 動力学の基本法則は 17世紀にニュートンによって見出された. それは物体の加速度と物体に働く力の比例関係を表すものであり, 物体の座標を時間の関数と見なすとき, 2階までの導関数を含む常微分方程式になる. さらに時代が下って, 19世紀に電気・磁気・熱現象が研究対象になると, 時間と空間の両方に関する物理量の変化の法則を書き下すために偏微分方程式も用いられるようになった.

今日では, 理工系の学問を学ぶ人にとって(程度の差はあっても)微分方程式に関する知識は必要不可欠である. さらに, 近年は経済学においても微分方程式の重要性が高まっている. もちろん, 純粋に数学的な観点から見ても, 微分方程式は重要かつ興味深いものであり, 解析学の中で中心的な位置を占めている.

- 簡単な問題を除き、解析解を求めるのは困難
  -> 数値解法が必要不可欠

1段法と多段法　p.154

•f(x, y)は　a≦ｘ≦b,　y∈ℜで定義された2変数関数とする．1階の**常微分方程式**　y’=f(x, y)　を考える．このとき，任意の　x∈　[a, b]に対して　y’=f(x, y(x))となるようなC[1]級の関数y=y(x)を常微分方程式の**一般解**または**解**という．y_0が与えられたとき，y(a)=y_0　を満たすような解を求める問題を常微分方程式の**初期値問題**という．

$$y' = f(x, y)$$
$$y(a) = y_0$$

•数値解法

- １段階法
- 多段階法

1段階法と多段階法　p.154

- 区間[a, b]を等間隔 $a=x_0<x_1<x_2<\cdots<x_n=b$, $h=(b-a)/n$ に分割

  $\phi(x,y)$ は与えられた関数，関数値$y(x_i)$の数値解をY_iとする．

- １段階法

$$\begin{aligned} Y_0 &= y_0 \\ Y_{i+1} &= Y_i + h\phi(x_i, Y_i),\ i = 0, 1, 2, \cdots, n-1 \end{aligned}$$

- 多段階法　　定数 α_j (j=0, 1, …, k−1)

$$\begin{aligned} Y_0 &= y_0 \\ Y_{i+1} &= \alpha_0 Y_i + \alpha_1 Y_{i-1} + \cdots + \alpha_{k-1} Y_{i-k+1} \\ &\quad + h\phi(x_i, x_{i-1}, \cdots, x_{i-k+1}, Y_i, Y_{i-1}, \cdots, Y_{i-k+1}, h),\ i = 0, 1, 2, \cdots, n-1 \end{aligned}$$

右辺にY_{i+1}が含まれている公式：陰公式

いない公式：陽公式

1段階法と多段階法　p.155

- １段階法

$$y_{i+1} = y_i + h\phi(x_i, y_i) + O(h^{p+1})$$

  - 次数pの精度を持つ，またはp次の精度をもつという．
  - 局所離散化誤差

$$\tau_i = \frac{y_{i+1} - y_i}{h} - \phi(x_i, y_i)$$

- 多段階法

$$\begin{aligned} y_{i+1} &= \alpha_0 y_i + \alpha_1 y_{i-1} + \cdots + \alpha_{k-1} y_{i-k+1} \\ &\quad + h\phi(x_i, x_{i-1}, \cdots, x_{i-k+1}, Y_i, Y_{i-1}, \cdots, Y_{i-k+1}, h) + O(h^{p+1}) \end{aligned}$$

  - 次数pの精度を持つ，またはp次の精度をもつという．
  - 局所離散化誤差

$$\begin{aligned} \tau_{i+1} &= \frac{y_{i-1} - (\alpha_0 y_i + \alpha_1 y_{i-1} + \cdots + \alpha_{k-1} y_{i-k+1})}{h} \\ &\quad - \phi(x_i, x_{i-1}, \cdots, x_{i-k+1}, y_i, y_{i-1}, \cdots, y_{i-k+1}, h) \end{aligned}$$

オイラー法　p.155

- [a,b]を n 等分し，各分点 x_i を

$$h = \frac{b-a}{n},\ x_0 = a,\ x_i = a + ih\ (i = 1, 2, \cdots, n)$$

h : 刻み幅

$y_i = y(x_i)$ とすると，テイラーの定理より

$$\begin{aligned} y_{i+1} &= y(x_i) + hy'(x_i) + \frac{h^2}{2}y''(x_i + \theta h),\ 0 < \theta < 1 \\ &= y_i + hf(x_i, y_i) + O(h^2) \end{aligned}$$

- オイラー法 （1次の精度を持つ1段階法）

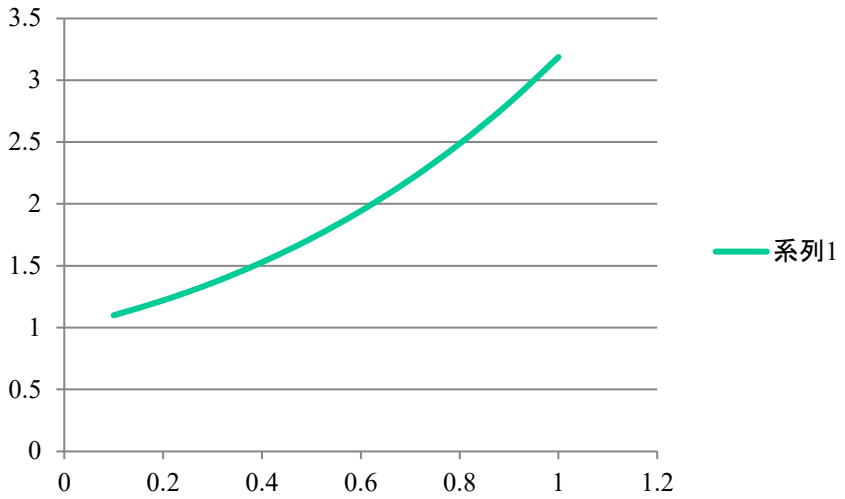
$$\begin{aligned} Y_0 &= y_0 \\ Y_{i+1} &= Y_i + hf(x_i, Y_i),\ i = 0, 1, 2, \cdots, n-1 \end{aligned}$$

オイラー法:プログラム p.157

```
#include <stdio.h>

/* 関数の定義 */
double func(double x, double y);
/* オイラー法 */
void euler(double x, double y, double a, double b, int n,
                        double (*f)(double, double) );

int main(void)
{
  int n;
  printf("分割数を入力してください--->");
  scanf("%d",&n);
  euler( 0.0, 1.0, 0.0, 1.0, n, func );
  return 0;
}

/* オイラー法 */
void euler(double x, double y, double a, double b, int n,
                        double (*f)(double, double) )
{
  double h;
  int i;

  h = (b-a)/n;  /* 刻み幅 */
  for ( i = 0 ; i < n ; i++) {
    y = y + h * (*f)(x,y);
    x += h;
    printf("x=%f ¥t y=%f ¥n", x, y );
  }
}

/* 関数の定義 */
double func(double x, double y)
{
  return( x + y );
}
```

実行結果

```
分割数を入力してください---> 10
x=0.1 y=1.1
x=0.2 y=1.22
x=0.3 y=1.362
x=0.4 y=1.5282
x=0.5 y=1.72102
x=0.6 y=1.943122
x=0.7 y=2.197434
x=0.8 y=2.487178
x=0.9 y=2.815895
x=1   y=3.187485
```



ホイン法　p.158

•式(8.2)より

$$y''(x) = \frac{d}{dx} f(x, y(x)) = \frac{\partial}{\partial x} f(x, y(x)) + \frac{\partial}{\partial y} f(x, y(x)) y'$$

2変数のテイラーの定理より

$$\begin{aligned} f(x + ah, y + \beta h) &= f(x, y) + \alpha h \frac{\partial f(x, y)}{\partial x} + \beta h \frac{\partial f(x, y)}{\partial y} + O(h^2) \\ f(x_i + h, y_i + h f(x_i, y_i)) &= f(x_i, y_i) + h \frac{\partial f(x_i, y_i)}{\partial x} + h f(x_i, y_i) \frac{\partial f(x_i, y_i)}{\partial y} + O(h^2) \\ &= f(x_i, y_i) + h y''(x_i) + O(h^2) \end{aligned}$$

1変数のテイラーの定理より

$$y_{i+1} = y_i + h y'(x_i) + \frac{h^2}{2} y''(x_i) + O(h^3)$$

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

ホイン法　p.158

•次数2の1段階法　　ホイン法　（Heun method）

$$
\begin{aligned}
Y_{i+1} &= Y_i + h\phi(x_i, Y_i),\ i = 0, 1, 2, \cdots, n-1 \\
\phi(x_i, Y_i) &= \frac{1}{2}(f(x_i, Y_i) + f(x_{i+1}, Y_i + hf(x_i, Y_i)))
\end{aligned}
$$

•ホイン法アルゴリズム

Input x, Y, a, b, n

$h \leftarrow (b-a)/n$

for $i = 0, 1, 2, \cdots, n-1$

$k_1 \leftarrow f(x, Y)$

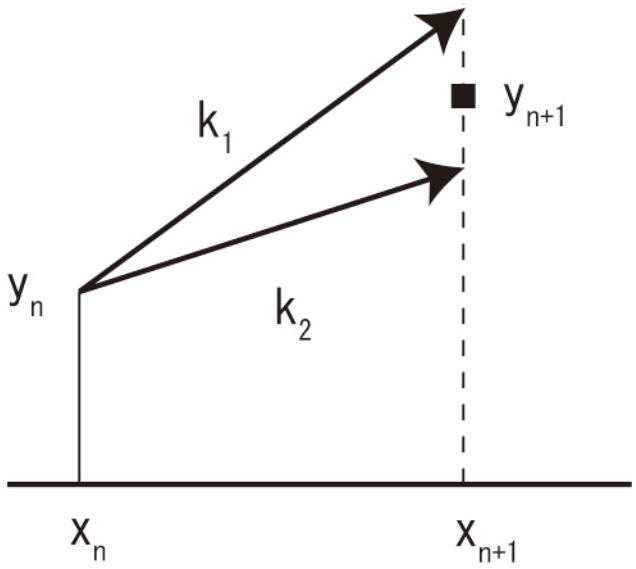
$k_2 \leftarrow f(x+h, Y+hk_1)$

$Y \leftarrow Y + \frac{h}{2}(k_1 + k_2)$

$x \leftarrow x + h$

Output Y

end for

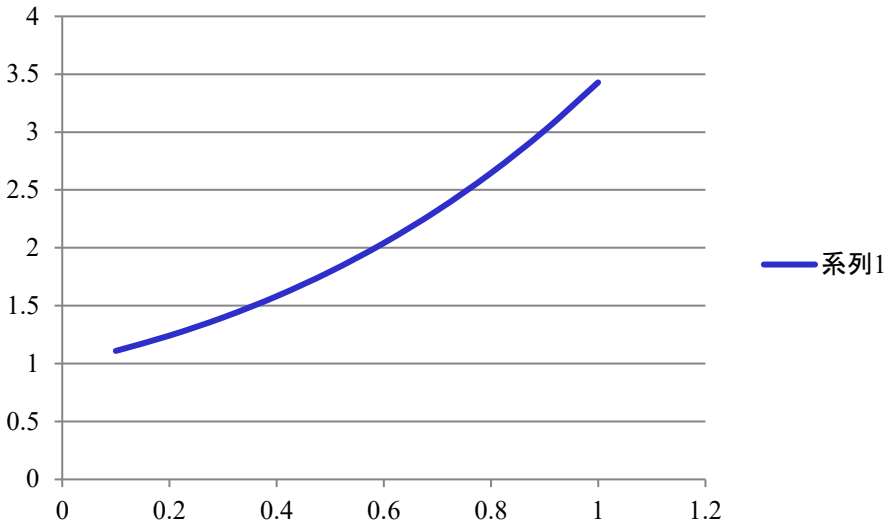


ホイン法:プログラム p.159

```
#include <stdio.h>

/* 関数の定義 */
double func(double x, double y);
/* ホイン法 */
void heun(double x, double y, double a, double b, int n,
                        double (*f)(double, double) );

int main(void)
{
  int n;
  printf("分割数を入力してください--->");
  scanf("%d",&n);
  heun( 0.0, 1.0, 0.0, 1.0, n, func );
  return 0;
}

/* ホイン法 */
void heun(double x, double y, double a, double b, int n,
                        double (*f)(double, double) )
{
  double k1, k2, h;
  int i;
  h = (b-a)/n;
  for ( i = 0 ; i < n ; i++){
    k1 = f(x,y); k2 = f(x+h, y+h*k1);
    y = y + h/2.0 * ( k1 + k2 );
    x += h;
    printf("x=%f ¥t y=%f ¥n", x, y );
  }
}

/* 関数の定義 */
double func(double x, double y)
{
  return( x + y );
}
```

実行結果

```
分割数を入力してください---> 10
x=0.1       y=1.11
x=0.2       y=1.24205
x=0.3       y=1.398465
x=0.4       y=1.581804
x=0.5       y=1.794894
x=0.6       y=2.040857
x=0.7       y=2.323147
x=0.8       y=2.645578
x=0.9       y=3.012364
x=1         y=3.428162
```



まとめ

- 常微分方程式
  - 1段階法と多段階法
  - オイラー法
  - ホイン法