科学計算研究室 Pythonゼミ

～12. 偏微分方程式 その2～

2021-04-19 福田　浩

1　原理(再掲)

偏微分方程式は，下記のようにして解くことが出来る．まず，微小区間 h に対して $u_{i,j}$ の前後の格子点上の関数値 $u_{i-1,j}$, u_{i+1}, j を用いて差分近似を考える．

$$\left(\frac{\partial u}{\partial x}\right)_{i,j} = \frac{u_{i,j} - u_{i-1,j}}{h} \tag{1}$$

$$\left(\frac{\partial u}{\partial x}\right)_{i,j} = \frac{u_{i+1,j} - u_{i,j}}{h} \tag{2}$$

$$\left(\frac{\partial u}{\partial x}\right)_{i,j} = \frac{u_{i+1,j} - u_{i-1,j}}{2h} \tag{3}$$

式 1 を後退差分近似，式 2 を前進差分近似，式 3 を中央差分近似と呼ぶ．

2 次の偏導関数は，さらにこの考え方を進めて，

$$\left(\frac{\partial^2 u}{\partial x^2}\right)_{i,j} = \frac{(\partial u/\partial x)_{i,j} - (\partial u/\partial x)_{i-1,j}}{h} \tag{4}$$

$$= \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} \tag{5}$$

で示すことが出来る．

2　双曲型偏微分方程式

$$\frac{\partial^2 u}{\partial t^2} = a\frac{\partial^2 u}{\partial x^2} \tag{6}$$

のタイプの偏微分方程式を，双曲型偏微分方程式と呼ぶ．代表的な例として．Maxwell 方程式がある．真空(または電荷分布が無い絶縁体中)での 1 次元の Maxwell 方程式は，下記の通りである．

$$\frac{\partial E}{\partial x} + \frac{\partial B}{\partial t} = 0 \tag{7}$$

$$\frac{\partial B}{\partial x} - \mu\epsilon\frac{\partial E}{\partial t} = 0 \tag{8}$$

ここで，E は電場, B は磁場，ϵ は誘電率，μ は透磁率である．式 7 を更に x で偏微分すると

$$\frac{\partial}{\partial x}\left(\frac{\partial E}{\partial x}\right) + \frac{\partial}{\partial x}\left(\frac{\partial B}{\partial t}\right) = 0 \tag{9}$$

$$\frac{\partial^2 E}{\partial x^2} + \frac{\partial}{\partial x}\left(\frac{\partial B}{\partial t}\right) = 0 \tag{10}$$

$$\frac{\partial^2 E}{\partial x^2} + \frac{\partial}{\partial t}\left(\frac{\partial B}{\partial x}\right) = 0 \tag{11}$$

ここで式 8 を変形して，式 11 に代入すると

$$\frac{\partial^2 E}{\partial x^2} + \mu\epsilon\frac{\partial^2 E}{\partial t^2} = 0 \tag{12}$$

となり，双曲型の偏微分方程式であることがわかる．

これを差分化すると，

$$
\begin{aligned}
\frac{E_{i+1}^{N} - 2E_{i}^{N} + E_{i+1}^{N}}{\Delta x^2} &= \mu\epsilon\left\{\frac{E_{i}^{N+1} - 2E_{i}^{N} + E_{i}^{N-1}}{\Delta t^2}\right\} &\quad (13)\\
\frac{E_{i}^{N+1} - 2E_{i}^{N} + E_{i}^{N-1}}{\Delta t^2} &= \frac{1}{\mu\epsilon}\left\{\frac{E_{i+1}^{N} - 2E_{i}^{N} + E_{i+1}^{N}}{\Delta x^2}\right\} &\quad (14)\\
E_{i}^{N+1} &= 2E_{i}^{N} - E_{i}^{N-1} + \frac{\Delta t^2}{\mu\epsilon}\left\{\frac{E_{i+1}^{N} - 2E_{i}^{N} + E_{i+1}^{N}}{\Delta x^2}\right\} &\quad (15)
\end{aligned}
$$

ここで上の添え字は時間 t を表し，下の添え字は位置 x を表す．

3 課題

Maxwell 方程式の数値解法プログラムを Python で実装せよ

- (簡単のために) 空間と時間は無次元化
- 全長 300 の 1 次元空間
- 時間は 200 単位時間まで計算
- 電場振幅と磁場振幅は規格化
- 電磁波源は 1 次元空間の中央に配置し， 40 単位時間で最大値になる単一パルス

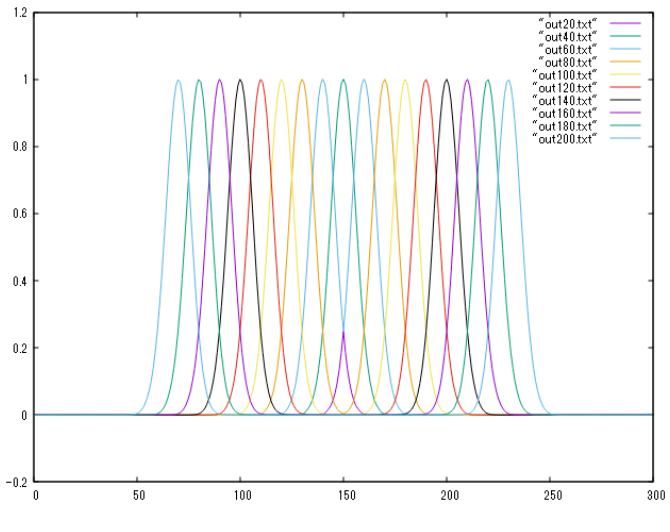

Figure 1: 課題の解答イメージ．

- 中央部分からパルスが立上り
- t=40 でパルス振幅が最大化
- 以降は左右に分かれて伝搬

4　参考：C++ソースコード (電場と磁場を順次計算する場合)

```
#include <iostream>
#include <math.h>
#define N 300
using namespace std;

int main(void){

    double ex[N], hy[N];
    double T;

    cin >> T;

    for(int i=0; i<N; i++){
        ex[i] = 0;
        hy[i] = 0;
    }

    for(int t=1;t<T+1;t++){
        for(int i=1; i<N; i++){
            ex[i] = ex[i] + (hy[i-1] - hy[i])/2;
        }
        ex[N/2] = exp(-0.5*(40-t)*(40-t)/144);
        for(int i=0; i<N-1; i++){
            hy[i] = hy[i] + (ex[i] - ex[i+1])/2;
        }
    }
    for(int i=0; i<N; i++){
      cout << i << " " << ex[i]  << endl;
    }
    return 0;
}
```

5 参考：C++ソースコード (電場だけを計算する場合)

```
#include <iostream>
#include <math.h>
#define N 300
using namespace std;

int main(void){

    double ex_old[N], ex_now[N], ex_new[N];
    double T;

    cin >> T;

    for(int i=0; i<N; i++){
        ex_old[i] = 0;
        ex_now[i] = 0;
        ex_new[i] = 0;
    }

    for(int t=1;t<T+1;t++){
        ex_now[N/2] = exp(-0.5*(40-t)*(40-t)/144);
        for(int i=2; i<N-1; i++){
            ex_new[i] = 2*ex_now[i] - ex_old[i]
            + (ex_now[i-1] - 2*ex_now[i] + ex_now[i+1])/4;
        }
        for(int i=1; i<N; i++){
            ex_old[i] = ex_now[i];
            ex_now[i] = ex_new[i];
        }
    }
    for(int i=0; i<N; i++){
      cout << i << " " << ex_now[i]  << endl;
    }
    return 0;
}
```