数値計算法概論：No.6 数値積分 (Richardson 加速)

1　数値積分の収束加速法

数値積分で台形公式を使って、刻み幅を小さくして精度をあげようとすると計算量が増大する。又、無闇に刻み幅を小さくすると今度は丸め誤差の影響がでてくる。そこで刻み幅の荒いデータから収束性を早くする方法を考えよう。

1.1　Euler-Maclaurin 総和公式

$f(x)$ が滑らかな関数の時、次のオイラー・マクローリン (Euler-Maclaurin) 総和公式が成立する [1, 2]。

$$\begin{aligned}
\int_a^b f(x)dx &= h\left[\frac{1}{2}f(a)+\sum_{i=1}^{i=N-1} f(a+hi)+\frac{1}{2}f(b)\right] \\
&- \frac{B_2h^2}{2!}(f'(b)-f'(a))-\cdots-\frac{B_{2k}h^{2k}}{(2k)!}(f^{(2k-1)}(b)-f^{(2k-1)}(a))-\cdots, \\
h &\equiv \frac{(b-a)}{N} \qquad (1)
\end{aligned}$$

ここで B_{2k} は Bernoulli 数と呼ばれ、次の関係で定義される。

$$\frac{t}{e^t-1}=\sum_{n=0}^{\infty} B_n\frac{t^n}{n!} \qquad (2)$$

具体的には

$$B_0=1, B_2=\frac{1}{6}, B_4=-\frac{1}{30}, B_6=\frac{1}{42}, \cdots \qquad (3)$$

(奇数値は $B_1=-1/2$ を除くとすべて消える)。

1.2　収束性

分割数 N の時の台形公式による数値積分の結果を I_N とする。

$$I_N = h_N\left[\frac{1}{2}f(a)+\sum_{i=1}^{i=N-1} f(a+h_Ni)+\frac{1}{2}f(b)\right];\ h_N=\frac{b-a}{N} \qquad (4)$$

台形公式の収束性はオイラー・マクローリンの公式 (1) で与えられ、h の偶関数である。分割数 N に対して誤差は $N^{-2}, N^{-4}, \ldots$ と振舞う。

$$I_N = I_\infty + c_1 N^{-2} + c_2 N^{-4} + c_3 N^{-6} \cdots \tag{5}$$

ところで I_N と I_∞ の直接比較して収束性評価できるのは結果が解析的に分かっている場合のみである。一般には、$I_{2N} - I_N$ を調べることで収束性を調べられる。例えば誤差が、

$$I_N = I_\infty + cN^{-\alpha} \tag{6}$$

と冪乗的に振舞うなら、$I_{2N} - I_N = c'N^{-\alpha}, \quad c' = c(2^{-\alpha} - 1)$ となる。さらに

$$(I_{4N} - I_{2N})/(I_{2N} - I_N) = 2^{-\alpha} \tag{7}$$

として誤差の収束の早さも評価できる。

1.3 台形公式の精度向上法:Richardson 補外

さらにオイラー・マクローリンの公式 (1) を使うと収束性を改善できる。つまり、

$$I_N = I_\infty + c_1 N^{-2} + c_2 N^{-4} + c_3 N^{-6} \cdots$$

と

$$I_{2N} = I_\infty + c_1 (2N)^{-2} + c_2 (2N)^{-4} + c_3 (2N)^{-6} \cdots$$

を比較して、

$$I_N^{(1)} = \frac{I_{2N} - I_N/4}{1 - 1/4} \tag{8}$$

とすると N^{-2} の誤差の項が打ち消されるので、

$$I_N^{(1)} - I_\infty = -\frac{c_2}{4} N^{-4} - \frac{5c_3}{16} N^{-6} \cdots \tag{9}$$

となり元の I_N より収束が早くなる。この操作を Richardson 加速 [3, 4] と呼ぶ。台形公式による数値積分を Richardson 加速すると、誤差は一般に N^{-4} のように振舞う。

上のようにして得られた $I_N^{(1)}$ の数列を、さらに

$$I_N^{(2)} = \frac{I_{2N}^{(1)} - I_N^{(1)}/16}{1 - 1/16} \tag{10}$$

とすると、誤差が N^{-6} のように振舞う。

特に分割数 N を $2, 4, 8, 16, 32, \cdots$ と 2 の冪乗 $N = 2^m$ にとると、

$$I_{2^m} = I_\infty + \sum_{j=1}^{\infty} c_j 2^{-2jm} \tag{11}$$

であるので、上の操作を繰り返すことができて、

$$I_N^{(k)} = \frac{I_{2N}^{(k-1)} - I_N^{(k-1)}/4^k}{1 - 1/4^k} \tag{12}$$

とするとさらに収束を早くできる。

2　プログラム

積分の上限、下限と分割数を与えた時、台形公式で数値積分の結果を出すサブルーチンを利用すると、分割数を 2 の冪乗とした時の結果を出すプログラムは以下のようになる。

```
c     Main routine
c
      implicit real*8 (a-h,o-z)
      dimension dInt(10,3)
c
      do m=1,6
         n=2**m
         dInt(m,1)=dIntegral(n,a,b)
      end do
c
      do m=1,6-1
         write(*,*) 2**m, dInt(m,1),dInt(m+1,1)-dInt(m,1)
      end do
      end
c
c
c     Integral routine(trapezoidal integral)
c
      real*8 function dIntegral(n,a,b)
      implicit real*8 (a-h,o-z)
      integer n,i
c
      ...........
      dIntegral=......
      return
      end
```

また、これを利用すると、Richardson 補外のプログラムができる。複数回 Richardson 補外を繰り返すには、subroutine 文を使うと有効である。

3　課題

(1-1) No.5 の積分に付いて、分割数 N を変えて I_N を計算し $O(h^2)(=O(1/N^2))$ の誤差の収束性を確かめよ。この中で、収束性が予期されるものよりずっと遅いものがある。その原因を考察して見よ。

(1-2) さらに Richardson 補外をして $I_N^{(1)}, I_N^{(2)} \ldots$ の収束性を見よ。以上のものの内、収束性が予想より早いものと、ずっと遅いものとがある。その原因を考察して見よ。

参考文献

[1] 「Numerical Recipes in C」、技術評論社、第 4 章 2 節

[2] 「数値計算法の数理」、杉原 正顕、室田 一雄 著、岩波書店、第 12 章 3 節

[3] 「数値計算の常識」、伊理 正夫、藤野 和建 著、共立出版、第 7 章

[4] 「数値計算法の数理」、杉原 正顕、室田 一雄 著、岩波書店、第 12 章 4 節