演習2 第4回 加速法

Yuta SAWA

2007年12月4日

1 加速法と補外法

数値計算においては、ある種の極限を取りたいことがある。例えば

$$\lim_{n\to\infty}\sum_{m=1}^{n}\frac{1}{2^m}=1 \tag{1}$$

のように、無限級数列の和を求めるというケースはその典型例である。この計算を有限項で打ち切る場合には、打ち切り誤差の影響が大きくなる。

またあるいは、数値微分の計算として片側差分法

$$f'(x)=\lim_{h\to 0}\frac{f(x+h)-f(x)}{h} \tag{2}$$

を考えると、この手法を用いる場合はある程度の大きさをもった h を用いて計算するしかない。

このようなときに有効になるのが補外を用いた**加速法**というテクニックである。加速法も様々なものがありそれらに対する使い方も多いが、今回は特に良くつかわれる Richardson 加速と Aitken 加速の 2 つ、及びそれらに対する多段加速法を説明する。またその発展形として、多項式補外と Euler 変換を簡単に紹介する。

1.1 Richardson 加速

1.1.1 Theory

ある数列 $\{a_n\}(n=1,2,3,\cdots)$ があり、この数列が定数 a に収束するとする。すなわち、$\lim_{n\to\infty}\{a_n-a\}=0$ を満たすとする。$|a_n-a|$ は打ち切り誤差を意味すると考えることもできる。

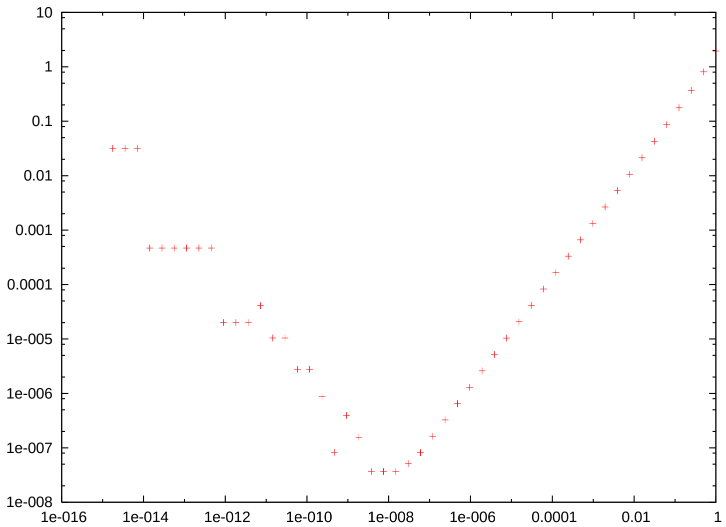

図 1: 加速無しの 1 次風上差分法の誤差

この誤差の**収束率**という概念を定義しよう。収束率 λ は

$$\lambda = \lim_{n\to\infty} \frac{a_{n+1} - a}{a_n - a} \tag{3}$$

と定義する。もしこの値 λ がある定数で、かつ事前に分かっていれば、

$$\begin{aligned} \lambda(a_n - a) &= a_{n+1} - a \\ a &\simeq \frac{a_{n+1} - \lambda a_n}{1 - \lambda} \end{aligned} \tag{4}$$

となる。これを **Richardson 加速法**と呼ぶ。

また、このような $lambda < 1$ が存在するとき、この数列は**指数収束する**という。

1.1.2 具体例

Richardson 加速は、基本的に事前に収束率がわかっていないと使うことはできない。例えば、式 (2) であらわされている 1 次風上差分法の誤差は

$$\frac{f(x+h) - f(x)}{h} - f'(x) = \frac{f''(x)}{2}h + \frac{f'''(x)}{6}h^2 + O(h^3) \tag{5}$$

と表現されるが、ここで、$a = f'(x)$, $a_n = \frac{f(x+h_n)-f(x)}{h_n}$ として、$h_n = 1/2^n$ とすれば、$h_{n+1}/h_n = 1/2$ となることより、

$$\frac{a_{n+1} - a}{a_n - a} = \frac{\frac{f''(x)}{2}h_{n+1} + \frac{f'''(x)}{6}h_{n+1}^2 + O(h_{n+1}^3)}{\frac{f''(x)}{2}h_n + \frac{f'''(x)}{6}h_n^2 + O(h_n^3)} \tag{6}$$

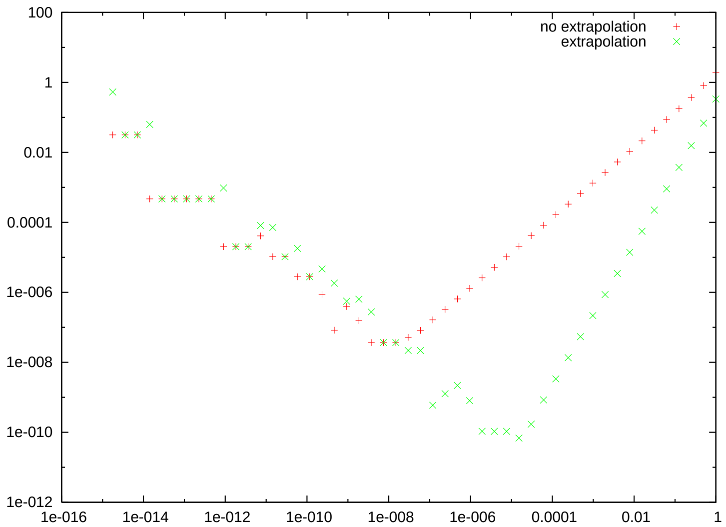

図 2: 加速有りの 1 次風上差分法の誤差

$$= \frac{1}{2} \frac{\frac{f''(x)}{2} + \frac{f'''(x)}{6} h_{n+1} + O(h_{n+1}^2)}{\frac{f''(x)}{2} + \frac{f'''(x)}{6} h_n + O(h_n^2)} \tag{7}$$

$$\to \frac{1}{2} \qquad (n \to \infty) \tag{8}$$

と出来る。これを用いて、関数 $f(x) = e^x$ における $x = 0$ の微分値 $f'(1) = e$ を数値的に計算をしてみよう。

```
#include <math.h>
#include <stdlib.h>
#include <stdio.h>

int main(){
  double h = 1.0;
  int i;
  double sol = exp(1);
  for(i = 0; i < 50; i++){
    double dif = (exp(1 + h) - exp(1)) / h;
    printf("%.20f %.20f %.20f\n", h, dif, fabs(dif-sol));
    h /= 2;
```

```
  }
}
```

この結果を図 1 に示す。右軸が h であり、縦軸が誤差である。この時、$h = 10^{-8}$ 辺りまでは誤差が小さくなっているが、そこから先は誤差が大きくなっている。これは丸め誤差によるものであるが、結局それ以上精度を上げることが出来ていない。

では、次にこれを加速したものを作ってみよう。$h = 1/2^n$ と $h = 1/2^{n+1}$ を用いて、式 (4) の加速を適用すると、図 2 のようになる。加速がない場合に比べて、より精度の高い解が、より荒い差分化で得ることが出来ていることがわかる。

なお、加速を用いたプログラムを記述すると

```
#include <math.h>
#include <stdlib.h>
#include <stdio.h>

int main(){
  double h = 1.0;
  int i;
  double sol = exp(1);
  for(i = 0; i < 50; i++){
    double a_n = (exp(1 + h) - exp(1)) / h;
    double a_np1 = (exp(1 + h/2) - exp(1)) / (h/2);
    double a = (a_np1 - 0.5 * a_n) / (1.0 - 0.5);
    printf("%.20f %.20f %.20f\n", h, a, fabs(a-sol));
    h/=2;
  }
}
```

というものである。

1.2 Aitken 加速

Richardson 加速は先にも述べたように収束率がわからないと使えないという欠点がある。従って、まず収束率を推定し、その収束率を使って Richardson 加速を行うという方法が Aitken 加速である。

収束率 λ を 2 つの方法で表現する。

$$\lambda = \frac{a_{n+1} - a}{a_n - a} \tag{9}$$

$$\lambda = \frac{a_{n+2} - a}{a_{n+1} - a} \tag{10}$$

これらから、まず収束値 a を左辺に持ってくると、

$$a = \frac{a_{n+1} - \lambda a_n}{1 - \lambda} \tag{11}$$

$$a = \frac{a_{n+2} - \lambda a_{n+1}}{1 - \lambda} \tag{12}$$

となり、これらからまず a を消去して

$$\frac{a_{n+1} - \lambda a_n}{1 - \lambda} = \frac{a_{n+2} - \lambda a_{n+1}}{1 - \lambda} \tag{13}$$

さらにこれから λ の値を推定すると、

$$\lambda = \frac{a_{n+2} - a_{n+1}}{a_{n+1} - a_n} \tag{14}$$

となる。

これを使うと、

$$a = a_n - \frac{(a_{n+1} - a_n)^2}{a_{n+2} - 2a_{n+1} + a_n} \tag{15}$$

と出来るので、収束率が不明であっても加速を適用できる。

1.3 多段加速法

今までの方法は、数列 a_n 誤差 $a_n - a$ が n に対して λ^n で減少していくというモデルであったが、もう少し複雑であることが多い。すなわち、誤差が複数存在していて、それらの誤差の収束率は異なることがほとんどである。そこで、誤差を

$$c_1\lambda_1^n + c_2\lambda_2^n + \cdots = \sum_i c_i\lambda_i^n \tag{16}$$

と出来るとする。この時に各 λ の値を $1 > \lambda_1 > \lambda_2 > \lambda_3 \cdots$ となるようにする。

そうすると、十分大きな n については $c_1\lambda_1^n >> c_2\lambda_2 >> c_3\lambda_3 >> \cdots$ となる。

1.3.1 多段 Richardson 加速

1 段の Richardson 加速の場合と同様に考える。例えば、2 段の加速を考えるとしよう。そうすると、ある程度大きい n についての誤差は、

$$a_n - a \simeq c_1\lambda_1^n + c_2\lambda_2^n \tag{17}$$

$$a_{n+1} - a \simeq c_1\lambda_1^{n+1} + c_2\lambda_2^{n+1} \tag{18}$$

$$a_{n+2} - a \simeq c_1\lambda_1^{n+2} + c_2\lambda_2^{n+2} \tag{19}$$

となり、ここから c_1 と c_2 を消去することで、

$$a = \frac{\lambda_1\lambda_2 a_n - (\lambda_1 + \lambda_2)a_{n+1} + a_{n+2}}{(1-\lambda_1)(1-\lambda_2)} \tag{20}$$

と書ける。これが 2 段の加速を行った Richardson 加速である。それ以上の段数の加速についてもまったく同じように消去ができる。

この手法を用いる場合、m 段の加速を行うには $m+1$ 個の a_n の値が必要である。

1.3.2 多段 Aitken 加速

多段 Aitken 加速についても考えてみよう。今回は、消去しなくてはならない変数が c_i 以外に λ_i も存在している。そうすると、例えば 2 段の加速を行うには

$$\begin{aligned}
a_n - a &\simeq c_1\lambda_1^n + c_2\lambda_2^n \\
a_{n+1} - a &\simeq c_1\lambda_1^{n+1} + c_2\lambda_2^{n+1} \\
a_{n+2} - a &\simeq c_1\lambda_1^{n+2} + c_2\lambda_2^{n+2} \\
a_{n+3} - a &\simeq c_1\lambda_1^{n+3} + c_2\lambda_2^{n+3} \\
a_{n+4} - a &\simeq c_1\lambda_1^{n+4} + c_2\lambda_2^{n+4}
\end{aligned} \tag{21}$$

のように、5 つのサンプル点が必要になる。この連立方程式を解くのは大変なので、まず (a_n, a_{n+1}, a_{n+2})、$(a_{n+1}, a_{n+2}, a_{n+3})$、$(a_{n+2}, a_{n+3}, a_{n+4})$ の 3 つの組を用いて Aitken 加速を行い、それぞれの結果を a'_n, a'_{n+1}, a'_{n+2} とすると、

$$a'_n = a_n - \frac{(a_{n+1} - a_n)^2}{a_{n+2} - 2a_{n+1} + a_n} \tag{22}$$

$$a'_{n+1} = a_{n+1} - \frac{(a_{n+2} - a_{n+1})^2}{a_{n+3} - 2a_{n+2} + a_{n+1}} \tag{23}$$

$$a'_{n+2} = a_{n+2} - \frac{(a_{n+3} - a_{n+2})^2}{a_{n+4} - 2a_{n+3} + a_{n+2}} \tag{24}$$

となる。今度はこれに Aitken 加速をさらに適用し、

$$a = a'_n - \frac{(a'_{n+1} - a'_n)^2}{a'_{n+2} - 2a'_{n+1} + a'_n} \tag{25}$$

としてやると、Aitken の 2 段の加速ができる。

1.4 その他の応用

1.4.1 Euler 変換

Euler 変換は、

$$S_n = \sum_{k=0}^{n} (-1)^k f_k \qquad f_k > 0 \tag{26}$$

として表現できるような数列の極限を求める場合に有効となる手法である。この場合、S_n が対数収束の場合に有効であることが多い。手法としては、上の S_n に対して $\lambda = -1$ として Richardson 加速を適用する。すなわち

$$S_n^{(1)} = \frac{S_{n+1} - (-S_n)}{1 - (-1)} = \frac{S_{n+1} + S_n}{2} \tag{27}$$

$$S_n^{(2)} = \frac{S_{n+1}^{(1)} - (-S_n^{(1)})}{1 - (-1)} = \frac{S_{n+1}^{(1)} + S_n^{(1)}}{2} \tag{28}$$

$$\vdots \tag{29}$$

である。要は、S_n に対して二つごとに平均を取っていけばよいということもできる。

またこれを整理すると

$$a_n^{(n+1)} = \frac{1}{2} \sum_{k=0}^{n} (-\frac{1}{2})^k \delta^k f_0 \tag{30}$$

ただし

$$\delta f_n = f_{n+1} - f_n \tag{31}$$

とあらわすことができる。

1.4.2 多項式補外

多項式補外は加速と同列に扱われることが多い。例えば先ほど Richardson 加速で見た数値微分のケースであれば、$n \to \infty$ というのと同様に差分 $h = 1/n \to 0$ である。従っ

て、数列の無限大の極限を求めることと、差分 h の 0 の極限を求めることは同じように論じることができる。誤差が h の多項式 $a_1h + a_2h^2 + a_3h^3 + \cdots$ とあらわせると仮定すれば、n 個の h について実験を行うことで n-1 項を削除することが可能になる。

このの h の取り方にはいくつかの手法があり、$1/2^n$ という形の Romberg 列や $1/2^n$ と $1/3*2^n$ の組み合わせという Bulirsh 列、$1/n$ のような Harmonic 列等がよくつかわれる。

2 課題

以下の課題を解答せよ。但し、課題 4-3 は余力があれば行うこと。それぞれの課題では、パラメータを変化させて実験を行い、グラフにして提出を行うこと。

2.1 課題 4-1

Richardson 加速を用いて、以下の 2 つのうち片方を実装せよ。

2.1.1 (1)

差分化を $h_{n+1} = h_n/4$ として、$x = 1$ における $\exp(x)$ の微分値を求め、Richardson 加速を行い、図 2 の結果と比較せよ。また可能であれば多段加速についても実装せよ。

なお、このようにした場合は収束率が 1/2 ではないことに注意せよ。

2.1.2 (2)

$x = 1$ における $\exp(x)$ の微分値を、差分化スキームとして中央差分法

$$f'(x) \simeq \frac{f(x+h) - f(x-h)}{2h} \tag{32}$$

を用いて計算せよ。またこれに Richardson 加速を行って、誤差の推移を調べよ。

2.2 課題 4-2

Aitken 加速を用いて、以下の 2 つのうち片方を実装せよ。

2.2.1 (1)

Aitken 加速を用いて数列 $a_n = \sum_{k=0}^{n} \frac{(-1)^k}{2k+1}$ の極限値を求めよ。なお、$a_n \to \pi/4 (n \to \infty)$ である。また段数を増やすとどうなるか。

2.2.2 (2)

多段 Aitken 加速を用いても無限級数列 $a_n = \sum_{k=1}^{n} 1/k^2 \to \pi^2/6 \simeq 1.644934 (n \to \infty)$ の収束値が精度良く求まるわけではないことを確認せよ。また、この部分列 $a'_n = a_{2^n} = \sum_{k=1}^{2^n} 1/k^2$ に対して Aitken 加速を適用し、この場合には精度良く求まることを確認せよ。

2.3 課題 4-3

以下の 2 つを実装せよ。

2.3.1 (1)

Euler 変換を用いて総和

$$\log 2 = 1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \cdots \quad (33)$$

$$= \sum_{k=1}^{\infty} (-1)^{(k-1)} \frac{1}{k} \quad (34)$$

の極限値を求めよ。

2.3.2 (2)

多項式補外を用いて、

$$\int_0^1 \exp(x) dx \quad (35)$$

の値を求めよ。数値積分としては、等間隔格子で台形公式を用いて計算すればよい。その際、h の刻みとして Romberg 列、Bulirsh 列、Harmonic 列のうち、二つ以上について実験して精度や計算量の観点から比較せよ。

[参考文献]

数値計算のわざ, 二宮市三, 共立出版, 2006 年
数値計算の基礎, 藤野清次, サイエンス社, 1998 年
数値計算の常識, 伊理正夫 藤野和建, 共立出版, 1985 年
数値計算法の数理, 杉原正顕 室田一雄, 岩波書店, 1994 年