ハイパフォーマンス
コンピューティング 59－2
（1995．12．11）

Euler-Maclaurinの総和公式による無限級数の加速法

平山　弘
hirayama@mse.kanagawa-it.ac.jp
神奈川工科大学 機械システム工学科
〒243-02 神奈川県厚木市下荻野1030番地

コンピュータが使われる以前からEuler-Maclaurinの公式は、無限級数の計算方法として使われてきた。Euler-Maclaurinの公式には、微分係数が含まれているために、コンピュータ上の計算法としては、ほとんど使われていない。C++言語を使うと、関数の微分係数は、高精度で計算できるため、Euler-Maclaurinの公式は、有力な無限級数の加速法となる。

Acceleration Method for Infinite Series by Euler-Maclaurin Summation Formula

Hiroshi Hirayama
hirayama@mse.kanagawa-it.ac.jp
Kanagawa Institute of Technology

Before a computer was used, the Euler-Maclaurin formula was being used for the acceleration method of an infinite series. Because differential coefficients of a function was contained in this formula, it wasn't used so much for the calculation on the computer. When a C++ language is used, differential coefficients can be computed easily in the high precision. It gives the very effective acceleration methods.

1. はじめに

級数の和を計算する方法には、古くから知られている Euler-Maclaurin の総和公式がある。この方法は、コンピュータ出現以前から使われている方法であるが、コンピュータが発展するに従って、あまり使われなくなって来ている。級数の数値計算関係の専門書[9]でも、この方法は扱われていない。数値計算の書籍[7][10]では、この公式の説明はあるが、この公式を直接使って、コンピュータで計算していない。

この理由は、この公式の中に関数の微分係数が入っているためである。微分係数は、これまでのプログラミング言語では精度よく計算できないためと考えられる。C++言語[1]は、従来の言語と異なり、オペレータ・オーバーローデング機能があり、ユーザ定義のデータ型や関数だけでなく、演算子も定義できる。この機能を使い、Taylor 級数の演算を定義する。これを利用すると、プログラムされた多くの関数を Taylor 展開することができる。Taylor 展開の係数は、もとの関数の微分係数の定数倍であるから、これを利用して、関数の微分係数が高精度で得られる。この計算の中には、数値微分のような桁落ちが生じる部分が無いので、高精度で計算できる。この計算は、従来の言語(FORTRAN や C 言語など)で行うことは不可能ではないが、大変面倒なプログラムとなる。今までほとんど使われていない事実をみると実用的には不可能とみるべきであろう。

本論文では、C++言語の機能を使い、従来の言語では扱えなかった、Taylor級数の演算を定義し、それを利用して、無限級数の加速方法を論じる。また、C++言語を利用すると高精度計算[6]が簡単に計算することができるので、計算精度や収束状況を見るために高精度計算を行った。

2. Euler-Maclaurin の総和公式よる加速法

次のような無限級数

$$s=\sum_{k=0}^{\infty} f(k) \tag{2.1}$$

を効率的に計算する方法を考える。このような無限級数の和を求めるのに、次の Euler-Maclaurin の総和公式[7]がある。

区間$[0,m]$で正則な関数$f(x)$の$x=0$からmまでの和を求めるには、次のような式となる。

$$\sum_{x=0}^{m} f(x) \approx \int_0^m f(x)dx+\frac{1}{2}\{f(0)+f(m)\}+\sum_{p=1}^{N-1}\frac{B_{2p}}{(2p)!}\left\{f^{(2p-1)}(m)-f^{(2p-1)}(0)\right\} \tag{2.2}$$

ここで、B_{2p} は Bernoulli 数である。この級数は、一般には発散級数である。この級数を使うには、Bernoulli 数を含む項が最小になった所で計算するのを止める必要がある。このとき、(2.2)の式の右辺と左辺の差はそのときの最小項の大きさ程度ある。これを利用して級数を計算することができる。

式(2.2)でmを無限大にすると、0から無限大までの無限級数に対する公式となる。

$$\sum_{x=0}^{\infty} f(x) \approx \int_0^{\infty} f(x)dx+\frac{1}{2}f(0)+\sum_{p=1}^{N-1}\frac{B_{2p}}{(2p)!}f^{(2p-1)}(0) \tag{2.3}$$

ただし、無限級数(2.1)は収束すると仮定しているので、関数$f(x)$は、xが無限大のとき、関数値と微分係数は0になることを利用して簡単化している。

本論文では、$x=1$から∞までの無限級数を扱う。また、最初のn項までは、通常の方法で計算し、$n+1$項から∞までの級数を扱うので、(2.3)の式を次のように変形する。

$$\sum_{x=1}^{\infty} f(x) \approx \sum_{x=1}^{n} f(x)+\int_{n+1}^{\infty} f(x)dx+\frac{1}{2}f(n+1)+\sum_{p=1}^{N-1}\frac{B_{2p}}{(2p)!}f^{(2p-1)}(n+1) \tag{2.4}$$

この式を使って、無限級数の計算をする。ここで、精度良く計算するためには、積分区間

$[n+1,\infty)$の項と、$x=n+1$における微分係数を精度良く求める必要がある。

微分係数の入っている式を数値計算するには、これまでの言語では、数値微分を使ってきた。数値微分では、関数$f(x)$の微分を、次の式

$$\frac{df}{dx}\approx\frac{f(x+h)-f(x)}{h} \tag{2.5}$$

で近似して計算する。ここで、h は、小さな数値である。h は、より小さい方がより微分係数に近い値が得られるが、h を非常に小さくすると、(2.5)の分子は、非常に近い数値の引き算となり、桁落ちが生じ、計算精度が著しく悪くなる。このため、数値微分法では、微分係数の高精度計算は期待できない。この問題点を Taylor 級数を求める計算方法で回避することができる。

3. 数値計算法

以上で述べた内容を計算手順としてまとめておく。

（1）級数を何項計算するかを決める。この個数をnとする。最終的に計算する精度および級数の性質に依存する。ここで扱った例では20項程度で倍精度(10進15桁程度)の結果が得られる。
（2）式(2.4)の第1項の級数を計算する。
（3）式(2.4)の第2項の半無限区間積分を計算する。この積分法には、いろいろな計算方法があるがここでは、定評のある二重指数型数値積分法[3]を使った。
（4）関数を$x=n+1$でテイラー展開を行う。このとき、C++言語の機能を使う。この計算によって、関数の高階微分係数が得られる。
（5）式(2.4)の第3項の級数を計算する。項が十分小さくなるまで計算する。希望する精度に達する前に、項が増大しはじめた場合には、nを大きくしてもう一度計算する。

4. 数値例

ここでは、Smith and Ford[8]のテスト問題の 7 番を例にして、具体的に説明を行い、最後に Smith and Ford のすべてのテスト問題の結果を示す。

4.1 簡単な例

次の無限級数の値を計算することを考える。

$$s=\sum_{k=1}\left(k+\exp\left(\frac{1}{k}\right)\right)^{-\sqrt{2}} \tag{4.1}$$

この問題では、最初の 19 項の和を計算する。すなわち次の計算を行う。

このとき、s1 = 1.023333852 得られる。次に、積分

$$\int_{20}^{\infty}\left(x+\exp\left(\frac{1}{x}\right)\right)^{-\sqrt{2}}dx \tag{4.2}$$

を計算する。これで計算すると s2 = 0.6836646926 が得られる。次に、次のような補正を行う。

$$s=s1+s2+\frac{1}{2}f(20)-\sum_{p=1}^{N-1}\frac{B_{2p}}{(2p)!}f^{(2p-1)}(20) \tag{4.3}$$

このためには、関数の高階の微係数を計算するために x=20 で、テイラー展開を計算する。テイラー展開を 5 次の項まで示すと次のようになる。

(4.4)
$$1.3446\times10^{-2}-9.0094\times10^{-4}(x-20)+5.1404\times10^{-5}(x-20)^2$$
$$-2.7581\times10^{-6}(x-20)^3+1.4310\times10^{-7}(x-20)^4-7.2634\times10^{-9}(x-20)^5$$

実際の計算は、倍精度の計算で、29次まで計算している。

この係数を使って、補正すると、7回の補正で倍精度の範囲で収束する。

(4.5) $s = 1.713796735540301$

の結果が得られる。

4.2 Smith and Fordのテスト問題

Smith and Ford のテスト問題[8]は、級数加速法の標準的な問題であり、収束の遅い無限級数である。多くの級数加速法によって計算されている問題でもある。これを計算することによって、ここで提案している計算方法の有効性と他の手法との比較を行うことができる。

本方法で計算した結果を表1に示す。以下の計算では、最初の 20 項を級数として計算し、それに対する積分を二重指数関数型数値積分で計算し、テイラー級数の係数で最終的に補正している。

テストの問題の半分程度は、式をそのまま記すことによって、計算可能であるが、桁落ちの生じる部分では、その対策をしなければならない。たとえば、4、5、6番の問題では、k がある程度以上大きくなると、対数関数部分は著しく精度が落ちる。このため、関数

(4.6) $\log 1(x) = \log(1+x)$

を定義し、この関数を桁落ちなしで計算できるルーチンを準備して計算した。この関数を利用すると、4番は

(4.7) $$\sin\left(\frac{1}{k}\right)\log\left(\cos\left(\frac{1}{\sqrt{k}}\right)\right) = \sin\left(\frac{1}{k}\right)\log 1\left(-2\sin^2\left(\frac{1}{2\sqrt{k}}\right)\right)$$

5番、6番は、それぞれ

(4.8) $$\frac{1}{k}+\log\left(\frac{k-1}{k}\right) = \frac{1}{k}+\log 1\left(-\frac{1}{k}\right)$$

(4.9) $$\log\left(\frac{k+1}{k}\right)\log\left(\frac{k+2}{k+1}\right) = \log 1\left(\frac{1}{k}\right)\log 1\left(\frac{1}{k+1}\right)$$

6番の問題は、(4.8)式の形でも桁落ちが生じるので、さらに桁落ちが生じないようにプログラムしなければならない。二項係数は、ガンマ関数との関係式

(4.10) $$(-1)^{k+1}\binom{-\frac{1}{2}}{k-1} = \frac{\Gamma\left(k-\frac{1}{2}\right)}{\pi\Gamma(k)}$$

から、計算している。ガンマ関数の計算には、ガンマ関数の対数の漸近展開式を利用して計算している。13番以降の交代級数は、

(4.11) $g(k) = f(2k-1)+f(2k)$

となる級数の和を計算することによって、計算している。

本方法は、昔から使われている計算方法であり、非常に単純であり、わかりやすいこと、さらに制限が余りなく、線形演算であることを考えると従来の計算法以上の利点があると思われる。

5 高精度計算

C++言語で記述した場合、テイラー展開が容易にできるだけでなく、高精度計算も容易になる。

ここでは、すでに発表されている高精度計算ルーチン[6]を利用して、Smith and Ford の問題について、要求精度 70 桁で計算した。その結果を表2に示す。

表１　収束の遅い級数

No.	$f(k)$	N	計算値
1	$\frac{1}{k^2}$	7	1.644934066848227
2	$\frac{1+k^2+k^4}{k^2(1+k^4)}$	7	2.223411646515362
3	$\frac{2k-1}{k(k+1)(k+2)}$	7	0.7499999999999999
4	$\sin\left(\frac{1}{k}\right)\log\left(\cos\left(\frac{1}{\sqrt{k}}\right)\right)$	7	-0.852090754198728
5	$\frac{1}{k}+\log\left(\frac{k-1}{k}\right)$	7	0.5772156649015329
6	$\log\left(\frac{k+1}{k}\right)\log\left(\frac{k+2}{k+1}\right)$	7	0.6847247885631572
7	$\left(k+\exp\left(\frac{1}{k}\right)\right)^{-\sqrt{2}}$	7	1.713796735540301
8	$\binom{-\frac{1}{2}}{k-1}\frac{(-1)^{k+1}}{4k-3}$	6	1.311028777146061
9	$\frac{1}{k^3}$	6	1.202056903159594
10	$\frac{1}{k^{1.5}}$	7	2.612375348685489
11	$\frac{1}{k^{1.5}}+\frac{1}{k^2}$	7	4.257309415533715
12	$\frac{\log(k)}{k^2}$	7	0.9375482543158438
13	$\frac{1}{k}(-1)^{k+1}$	7	0.6931471805599451
14	$\frac{1}{2k-1}(-1)^{k+1}$	6	0.7853981633974481
15	$\frac{1}{\sqrt{k}}(-1)^{k+1}$	7	0.6048986434216305
16	$\frac{1}{k}\binom{-\frac{1}{2}}{k-1}$	6	0.8284271247461905
17	$\binom{-\frac{1}{2}}{k-1}^2(-1)^{k+1}$	6	0.8346268416740737

表２の計算結果は、式(2.4)において、n=100 および n=110 の場合を計算し、二つの結果が一致することを確かめている。Taylor 級数は、すべて 100 次まで計算してあるが、最初の 30 項弱で、す

べて問題が収束している。計算時間の大半が二重指数関数型数値積分の時間である。標本点数は、問題にかかわらず750から800個程度であった。

Pentium 133Mhz を使ったコンピューターで、計算時間を測定すると、一番の問題がもっとも短く、3.9秒で、最も時間のかかる問題が、8番の問題で127.8秒であった。二項係数を含む問題は100秒程度かかり、それ以外の問題は、10から20秒程度で計算できる。

表2　収束の遅い級数の高精度計算

No.	計算値
1	1.6449340668 4822643647 2415166646 0251892189 4990120679 8437735558 2293700075
2	2.2234116465 1536327479 0437359891 7444242656 2211853408 9765323044 8751638159
3	0.7500000000 0000000000 0000000000 0000000000 0000000000 0000000000 0000000000
4	-0.8520907541 9872795601 5117677248 7772819260 0986293976 7395859089 5610376394
5	0.5772156649 0153286060 6512090082 4024310421 5933593992 3598805767 2348848679
6	0.6847247885 6315712329 9146148755 7776204606 7541633744 8836606289 8678159569
7	1.7137967355 4030148654 2998791306 2625487486 1671673813 2713922257 4328733901
8	1.3110287771 4605990523 2419794945 5597068413 7747571581 1581408410 8519003953
9	1.2020569031 5959428539 9738161511 4499907649 8629234049 8881792271 5553418382
10	2.6123753486 8548834334 8567567924 0716305708 0065240006 3407573328 2488149277
11	4.2573094155 3371477982 0982734570 0968197897 5055360686 1845308886 4781849352
12	0.9375482543 1584375370 2574094567 8649778978 6028861482 9925885433 4803604438
13	0.6931471805 5994530941 7232121458 1765680755 0013436025 5254120680 0094933936
14	0.7853981633 9744830961 5660845819 8757210492 9234984377 6455243736 1480769541
15	0.6048986434 2163037024 7265914235 9554997597 6254513024 7380378546 6480821872
16	0.8284271247 4619009760 3377448419 3961571393 4375075389 6146353359 4759814649
17	0.8346268416 7407318628 1429732799 0468089939 9301349034 7002449827 3701036820

6 結論

Euler-Maclaurin の公式を利用した無限計算方法は、C++言語を利用するとコンピュータを利用した計算法としても利用できることを示した。この方法は、今まで開発されている加速法と同じ程度の性能を発揮する。また、C++言語を利用しているため、C++言語で開発されているいろいろなプログラムなどが容易に利用できる。ここでは、例として、高精度計算を行った。

以上の計算は、Borland C++ Ver. 4.5およびWatcom C/C++ 10.5Jを使って作成した。計算には、GATEWAY 2000 P5-133を利用した。

参考文献

[1] Ellis M. A. and Stroustrup B., The Annotated C++ Reference Manual, Addison-Wesley,1990

[2] 久保田光一, コンピューティングの玉手箱 続・自動微分 Taylor 展開,bit,Vol.22, No.5(1991), pp.860-861

[3] 森　正武, 数値解析と複素関数論, 筑摩書房, 東京, 1975

[4] Rall L. B., Automatic Differentiation Technique and Applications, Lecture Notes in Computer Science, Vol.120, Springer Verlag, Berlin-Heidelberg-New York,1981

[6] 平山　弘, C++言語による高精度計算パッケージの開発, 日本応用数理学会論文誌, Vol. 5, No.3 (1995), pp. 123-318

[7] Stanton R. G., Numerical Methods for Science and Engineering, Prentice-Hall, Englewood, 1961

[8] Smith D. A. and Ford W. F., Acceleration of Linear and Logarithmic Convergence, SIAM J. Num. Anal., Vol.16, No. 2, pp. 223-240(1979).

[9]Wimp J., Sequence Transformations and Their Applications, Academic Press, New York, 1981

[10]山本哲朗, 数値解析入門, サイエンス社, 東京, 1976