1O10 **相補誤差関数の最良近似：Hastings の近似式の再検討**

〇河野 明男
(独) 海洋研究開発機構　地球シミュレータセンター（〒 236-0001 横浜市金沢区昭和町 3173-25）

【序】

相補誤差関数　相補誤差関数は erfc と表記され，

$$\mathrm{erfc}(x) = 1 - \mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty \exp(-t^2)dt \tag{1}$$

で定義される超越関数であり（図 1），拡散問題や統計理論などの様々な分野で用いられる。分子シミュレーションでは erfc は Ewald 和において使われている。これまでに erfc の値を計算するための様々な近似が提案されている。

多くの $\mathrm{erfc}(x)$ の近似は $x \in [0, \infty)$ の範囲で動作するように作られている。実際，Ewald 和を含む多くの応用では引数 x は非負実数に限定されていることが多い。それでも負の引数における erfc の値が必要であれば，関係式

$$\mathrm{erfc}(-x) = 2 - \mathrm{erfc}(x) \tag{2}$$

を用いてそれを得ることができる。

区分的手法は erfc の値を得るためによく用いられている。これは定義域 $[0, \infty)$ を複数の区分に分割し，それぞれの区分で別個の近似式を用いるものである。このような方法は精度について有利となるが，その実装には条件分岐を伴うため，深いパイプラインや SIMD 命令を持つ近年のプロセッサ，ベクトルプロセッサ，GPGPU などを用いた計算において効率の低下をもたらす。そこで本研究では区分的手法は取り扱わない。

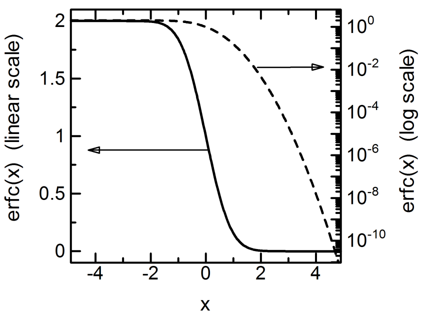

図 1: 相補誤差関数のプロット。

最良近似　試行近似関数のうち，その最悪の近似誤差の大きさが最も小さいものをミニマックス近似あるいは最良近似とよぶ。ある実数値関数 f を領域 $[a, b]$ で近似することを考える。点 $x \in [a, b]$ における近似関数 g の近似誤差を

$$e(x; g) = w(x)[g(x) - f(x)] \tag{3}$$

と定義する。ここで $w(x)$ は定符号の重み関数であり，$w(x) = 1$ のとき絶対誤差，$w(x) = 1/f(x)$ のとき相対誤差となる。近似誤差 e の大きさの最大値を E とする。すなわち

$$E[g] = \max_{x \in [a,b]} |e(x; g)|. \tag{4}$$

試行近似関数を $g_{\boldsymbol{c}}$ とする。ここで $\boldsymbol{c}$ は関数の形状を指定する実パラメータベクトルである。ここである実ベクトル $\boldsymbol{c}^*$ が任意の実ベクトル $\boldsymbol{c}$ について $E[g_{\boldsymbol{c}^*}] \le E[g_{\boldsymbol{c}}]$ を満たすとき，関数 $g_{\boldsymbol{c}^*}$ は最良近似である。

Hastings の近似式　Hastings は $\mathrm{erfc}(x)$ の定義域 $x \in [0, \infty)$ に対する近似式の一つとして

$$g(x) = \exp(-x^2)\, t(x)\, P_N[t(x)] \tag{5}$$

を提案した [1]。ここで $t(x) = 1/(1 + px)$, P_N は次数 N の多項式で $P_N(\xi) = a_0 + a_1\xi + \cdots + a_N\xi^N$, p は正の係数, $\{a_0, \ldots, a_N\}$ は実係数である。Hastings は絶対誤差に関して最良となる $N = 2, 3, 4$ の係数を与えた。これらの係数は拘束条件 $g(0) = \mathrm{erfc}(0)$ を満たすように決められた。この条件は式 (2) を用いて負の引数に対する近似値を評価する際に，$x = 0$ での値と一階微分の連続性を保証する。

この $N = 2, 4$ の近似式と係数は有名な Abramowitz と Stegun のハンドブック [2] に掲載されたため，erfc の近似として最も良く知られているものの一つとなり，多くの教科書や Wikipedia（英語版）にも引用されている。またこれは評価に $\exp(-x^2)$ の計算を伴う。Ewald 和の力の評価にも $\exp(-x^2)$ が用いられるため，Ewald 和と相性が良い。そのためこの近似は LAMMPS などの分子動力学法パッケージにも用いられている。

目的　ここで Hastings の近似式の問題点を指摘したい。

1. 係数は $N = 2, 3, 4$ の場合のみ与えられている。Hastings の与えた $N = 4$ の近似は絶対誤差の大きさが最大 1.39×10^{-7} であり，より正確な $N > 4$ の近似式が必要な場合もあるだろう。計算機においては除算や指数関数の計算は乗算や加算よりも高コストであるため，多少の N の増減では要する計算時間はほとんど変わらない。また，間に合わせの手軽な実装のためには $N = 1$ の近似式も有用であろう。

2. 係数は相対誤差ではなく絶対誤差の大きさを最小化するように決定されており，その相対誤差はかなりの大きさとなる。相補誤差関数は 0 に漸近する関数であるため，多くの場合は絶対誤差よりも相対誤差についての最良近似が望ましい。

3. 係数が真の最良近似を与えていることが確認されていない。一般に $E[g]$ には複数の極小が存在するため，単純なアルゴリズムによって得られた係数は局所最適であるが大域最適ではない可能性がある。

4. 最良な係数を求めるための Remez アルゴリズムは N が大きい時は収束させることが困難であり，式 (5) に対して Remez アルゴリズムの主要部分が非線形となるため，特に N が大きい場合，収束解を得るには適切な初期推測を与える必要がある。また収束したとしてもその係数は局所最適であって大域最適でない可能性がある。

本研究では，erfc に対する近似式 (5) の最良近似を与える係数を求めるロバストな手法を開発した。これを用いて $N = 1, 2, \ldots, 30$ について相対誤差，絶対誤差に関する最良な係数を決定した。さらに，Hastings が与えた，最も広く使われている $N = 4$ の係数は実は局所最適解の一つであって最良ではないことを示す。

【結果】

Hastings の近似式 (5) について最良な係数を決定するためのプログラムを作成した。任意精度浮動小数点演算ライブラリには GNU MPFR を用い，計算は 512 ビットの精度で行った。

絶対誤差に関する最良近似について最大絶対誤差と N の関係を図 2 に示す。相対誤差に関する最良近似について最大相対誤差と N の関係を図 3 に示す。

我々の得た $N = 2, 3$ の絶対誤差に関して最良な係数は Hastings によって与えられた係数と一致する。しかし，最も広く用いられている $N = 4$ の Hastings の係数は我々の得た最良の係数とは異なっており，局所最適解の一つとして得られた係数と一致する。我々の得た係数は Hastings のオリジナルの係数と比べて絶対誤差をわずかながら改善し，相対誤差を大幅に改善する（図 4）。

我々の得た $N = 4$ の係数を用いた絶対誤差に関する erfc の最良近似関数のサンプルコードを図 5 に示す。

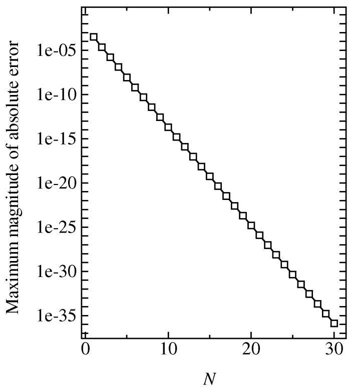

図 2: 絶対誤差に関する最良近似について最大絶対誤差と N の関係。

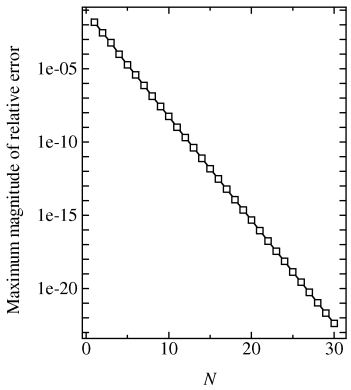

図 3: 相対誤差に関する最良近似について最大相対誤差と N の関係。

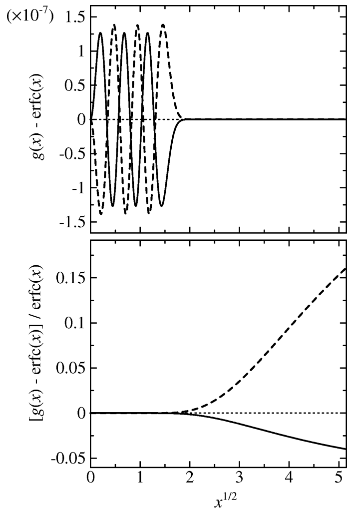

図 4: Hastings の係数と我々の得た係数を用いた近似誤差の比較。破線は Hastings の，実線は我々の係数を用いた近似誤差。上のパネルには絶対誤差，下のパネルには相対誤差を示す。

```
#include <math.h>
#define ERFC_APPROX_A0   0.24415132447597377699
#define ERFC_APPROX_A1   0.40888021168894839544
#define ERFC_APPROX_A2  -0.18289935763498826791
#define ERFC_APPROX_A3   0.75062257248685838204
#define ERFC_APPROX_A4  -0.22075486818349616103
#define ERFC_APPROX_P    0.46782803228416853550

double erfc_approx(double x)
{
    double t = (1.0/ERFC_APPROX_P) / ((1.0/ERFC_APPROX_P) + x);
    double y = ERFC_APPROX_A4;
    y = y * t + ERFC_APPROX_A3;
    y = y * t + ERFC_APPROX_A2;
    y = y * t + ERFC_APPROX_A1;
    y = y * t + ERFC_APPROX_A0;
    return y * t * exp(-x * x);
}
```

図 5: 我々の得た $N = 4$ の係数を用いた絶対誤差に関する erfc の最良近似関数のサンプルコード（C 言語）。

参考文献

[1] C. Hastings, Jr., *Approximations for Digital Computers*, Princeton University Press, Princeton (1955).

[2] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, Dover, New York (1964).