第8章 初等関数の計算法

かつてのDSPの応用は，主としてディジタル・フィルタでした．しかし現在では，DSPの応用が各方面に広がっています．そのため，平方根，三角関数，指数関数などの初等関数の計算も必要になる場合が出てきます．

C/C++言語でプログラムを開発する場合，浮動小数点演算では標準ライブラリを使って初等関数の値を簡単に計算できます．しかし，固定小数点演算の場合は汎用的なライブラリがあまりないため，自分でそのような関数を作らなければならないことがあります．また，浮動小数点演算の場合でも，標準のライブラリで提供されている関数は精度が高いので，それほど高い精度を必要とせず実行スピードの速い関数が必要になるということもあります．したがって，初等関数の計算方法をある程度知っておく必要があります．

本章では，関数近似の方法を紹介し，そのプログラムの実例を示します．

8.1 関数近似の方法

関数の近似を行う方法は，いろいろと知られています．ここでは，以下の方法について説明します．

1. ニュートン(Newton)法
2. 近似式の利用
3. CORDIC法
4. その他の方法

(1) ニュートン法

ニュートン法は，方程式を数値的に解く方法として有名な方法です．解こうとする方程式を$f(x)=0$とし，$f(x)$は微分可能であるとします．そのとき，$f(x)=0$を満足するxの近似値，つまり根の近似値を，次の式を繰り返すことによって求めるのがニュートン法です．

$$x^{(n+1)} = x^{(n)} - \frac{f(x^{(n)})}{f'(x^{(n)})}, \quad n = 0, 1, \cdots \quad \cdots\cdots (8.1)$$

この式で，初期値として$x^{(0)}$に適切な値を選び，式(8.1)の計算を繰り返していけば，$x^{(n+1)}$は$f(x) = 0$を満足するxに近づいていきます．

ニュートン法では，初期値$x^{(0)}$の与え方が問題になります．初期値が根の真値に近い場合は，式(8.1)を繰り返せば非常に急速に真値に近づくことが知られています[注8.1]．しかし，初期値$x^{(0)}$の与え方が適切でなければ正しい近似値が求められないこともあるので，使う場合は注意が必要です．

C67xでは，ニュートン法の初期値として使うために，逆数や平方根の逆数の近似値を求める命令がサポートされており，これをC/C++言語から直接使うことができます．これらの命令を使った例については，プログラムに関する節で示します．

次に，ニュートン法を適用するいくつかの例を示します．

● **逆数**

$1/a$は，次の方程式の根として求めることができます．

$$f(x) = x^{-1} - a = 0 \quad \cdots\cdots (8.2)$$

この方程式の根の近似値を求めるための，ニュートン法による反復の式は次のようになります．

$$x^{(n+1)} = x^{(n)}\left(2 - ax^{(n)}\right), \quad n = 0, 1, \cdots \quad \cdots\cdots (8.3)$$

● **平方根(その1)**

$\sqrt{a}$を求める場合，直接的には次の方程式を解くことになります．

$$f(x) = x^2 - a = 0 \quad \cdots\cdots (8.4)$$

この方程式の根の近似値を求めるための，ニュートン法による反復の式は次のようになります．

$$x^{(n+1)} = 0.5\left(x^{(n)} + \frac{a}{x^{(n)}}\right), \quad n = 0, 1, \cdots \quad \cdots\cdots (8.5)$$

● **平方根(その2)**

平方根(その1)では，反復の式の中に除算を含みます．ところで，DSPは一般に高速のハードウェア除算器を内蔵していないのが普通です．C6xシリーズのDSPも同様です．そのため，除算が入ってくると，実行スピードが遅くなってしまいます．

除算を使わない方法としては，最初に$1/\sqrt{a}$を求め，次に$\left(1/\sqrt{a}\right)\times a$で$\sqrt{a}$を求めるという方法があります．$1/\sqrt{a}$は，次の方程式の根になります．

$$f(x) = x^{-2} - a = 0 \quad \cdots\cdots (8.6)$$

この方程式の根の近似値をニュートン法により反復的に求めるためには，次の式を使います．

$$x^{(n+1)} = x^{(n)}\left(1.5 - 0.5a\left(x^{(n)}\right)^2\right), \quad n = 0, 1, \cdots \quad \cdots\cdots (8.7)$$

注8.1：式(8.1)をn回繰り返したときの誤差を$\varepsilon^{(n)}$とすると，ニュートン法では$|\varepsilon^{(n)}|$が十分小さければ

$$|\varepsilon^{(n+1)}| \cong c\left(\varepsilon^{(n)}\right)^2, \quad c\text{は定数}$$

となることが知られている．つまり，繰り返すごとに，誤差は前回の誤差のほぼ2乗程度に減少することになる．

(2) 近似式

近似式には，多項式近似，有理関数近似，連分数近似などがあります．この中で，有理関数や連分数を計算するためには除算が必要になるため，DSPに向いている方法とはいえません．DSPで計算する際は，多項式を使って近似する方法が適しています．

その際に使う近似多項式ですが，テーラー(Taylor)展開やマクローリン(Maclaurin)展開で求めることもできます．しかし，これらの展開で求めた多項式は，展開の中心から離れると誤差が急激に大きくなり，必要な精度を得るためには項の数を多く必要とします．そのため，計算時間も長くなってしまいます．

そのため，少ない項数でできるだけ誤差を小さくしたいという場合は，ミニマックス近似[注8.2]で求めた近似式がよく使われます．

● ミニマックス近似

一般に，近似式を求める場合は，ある限定された区間で考えます．そこで，ある関数$f(x)$の近似関数を$g(x)$とし，近似の区間を$a \le x \le b$とします．

ミニマックス近似とは，区間$a \le x \le b$で，誤差の絶対値の最大値が最小になるように近似する方法です．誤差には絶対誤差と相対誤差がありますが，絶対誤差で考えると次のように表すことができます．

$$\max_{a \le x \le b}\left\{\left|f(x)-g(x)\right|\right\} \rightarrow \text{最小化} \quad \cdots\cdots (8.8)$$

$\max_{a \le x \le b}\{\ \}$は，区間$a \le x \le b$での最大値を表します．

相対誤差で考えると，ミニマックス近似は次のようになります．

$$\max_{a \le x \le b}\left\{\left|\frac{f(x)-g(x)}{f(x)}\right|\right\} \rightarrow \text{最小化} \quad \cdots\cdots (8.9)$$

ミニマックス近似式には，誤差について次のような性質があります．ミニマックス近似式には誤差が極大になる点，および極小になる点が複数個ありますが，それらの絶対値はすべて等しいという性質です．この性質は，近似式の係数を逐次的に修正しながらミニマックス近似式を求める場合の指針になります．

ミニマックス近似式を求めるのは，それほど簡単ではありません．しかし，世の中にはミニマックス近似式をまとめた書物もあるので，そのようなものを参考にするのがよいでしょう．参考文献1)には，さまざまな関数について，色々な誤差のミニマックス近似式が載っています．

しかし，ミニマックス近似式の求め方に関心がある読者もいることと思います．また，参考文献1)では誤差の小さな近似式は載っていますが，誤差が大きな近似式は載っていない場合もあります．

ある関数のミニマックス近似式が解析的に求められる場合というのはあまりありません．そのため，普通は次の手順でミニマックス近似式を求めます．

① 最初に，ミニマックス近似式に近い第1近似式を求める．

注8.2：最良近似と呼ばれる場合もある．

② 誤差の極大値，極小値を求め，その絶対値が等しくなるように第一近似式の係数を逐次的に修正していく．

第1近似式を求めるためにはいくつかの方法がありますが，比較的簡単なチェビシェフ補間で近似式を求める方法について，**コラムG**に示します．第一近似式の係数を逐次的に修正してミニマックス近似式を求める方法については，**コラムH**に示します．

● **ミニマックス近似式の例（sin関数）**

使う近似式を決めるには，要求される精度を決める必要があります．ここではsin関数の値をshort型（16ビット）で表現するということで考えてみます．

コラムG　チェビシェフ補間

ミニマックス近似式を求める場合に使う第1近似式の求め方にはいくつかの方法がありますが，ここではチェビシェフ補間（Chebyshev interpolation）による方法を紹介します．

ある関数$f(x)$の第1近似式を，区間$[-1, 1]$で求めるものとします．近似式をN次の多項式$g(x)$とし，これをチェビシェフの多項式[注G.1] $T_n(x)$を使って表すと，次のようになります．

$$g(x) = a_0 T_0 + a_1 T_1 + \cdots + a_N T_N \quad \cdots\cdots (G.1)$$

チェビシェフ補間では，この係数a_0，a_1，$\cdots$，a_Nを以下のようにして求めます．

まず，r_kという値を次のように決めます．

$$r_k = \cos\frac{(2k+1)\pi}{2(N+1)} \quad \cdots\cdots (G.2)$$

このr_kを使うと，係数a_0，a_1，$\cdots$，a_Nは次の式で計算されます．

$$\begin{cases} a_0 = \dfrac{1}{N+1}\displaystyle\sum_{k=0}^{N} f(r_k), & n = 0 \\ a_n = \dfrac{2}{N+1}\displaystyle\sum_{k=0}^{N} f(r_k)\cos\dfrac{(2k+1)n\pi}{2(N+1)}, & n = 1, 2, \cdots, N \end{cases} \quad \cdots\cdots (G.3)$$

次に，式(G.3)で求められた係数とチェビシェフの多項式を式(G.1)に代入し，さらにこれを展開して同じ次数のxについてまとめると，xに関するN次の多項式$g(x)$が求められます．この式が第1近似式になります．

近似の区間が$[x_1, x_2]$の場合は，最初に次に示す変数変換を行います．

注G.1：チェビシェフの多項式（Chebyshev's polynomial）$T_n(x)$の一般的な定義は次のようになる．

$$T_n(x) = \begin{cases} \cos(n\cos^{-1}x), & |x| \le 1 \\ \cosh(n\cosh^{-1}x), & |x| > 1 \end{cases} \quad \cdots\cdots (a)$$

いくつかのnについて具体的な形で書くと次のようになる．

$$\left.\begin{array}{ll} T_0(x) = 1 & T_1(x) = x \\ T_2(x) = 2x^2 - 1 & T_3(x) = 4x^3 - 3x \\ T_4(x) = 8x^4 - 8x^2 + 1 & T_5(x) = 16x^5 - 20x^3 + 5x \\ T_6(x) = 32x^6 - 48x^4 + 18x^2 - 1 & T_7(x) = 64x^7 - 112x^5 + 56x^3 - 7x \end{array}\right\} \quad \cdots\cdots (b)$$

また，チェビシェフの多項式は次の関係式が成り立つので，高次のチェビシェフ多項式は，低次のものから再帰的に求めることができる．

$$T_{n+2}(x) = 2xT_{n+1}(x) - T_n(x), \quad n \ge 0 \quad \cdots\cdots (c)$$

sin関数の値の取り得る範囲は，−1～1です．そこで，1や−1を正確に表現しようとすると，符号ビットを含めて，小数点以上に2ビットを割り当てる必要があります．そうすると，sin関数の値をshort型で表現する場合，小数点以下には14ビット割り当てることになります．したがって，近似値の絶対誤差[注8.3]の絶対値が$2^{-14} \cong 6.10 \times 10^{-5}$よりも小さくなる近似式を選択します．参考文献1)のp.201にsin関数のミニマックス近似式が載っていますが，この条件に当てはまるものでもっとも

注8.3：誤差には，絶対誤差と相対誤差がある．絶対誤差とは真値−近似値で求められる値で，相対誤差とは(真値−近似値)/真値で求められる値である．

$$u = \frac{2x - (x_2 + x_1)}{x_2 - x_1} \quad \cdots\cdots (G.4)$$

これを利用して，関数$f(x)$をuのチェビシェフ多項式で近似します．次に，uに関する多項式にまとめ，最後に，式(G.4)を考慮して区間$[x_1, x_2]$におけるxについての近似式に変換します．

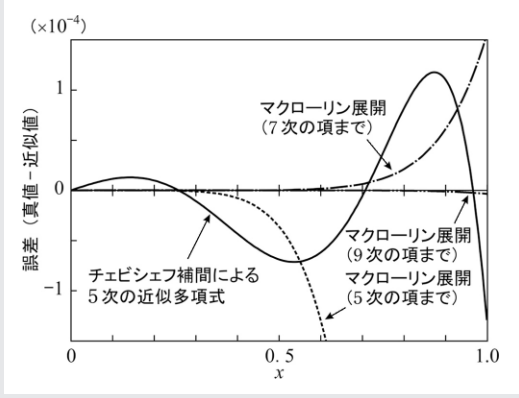
例として，チェビシェフ補間で求めた，区間$[-1, 1]$における$\sin(\pi x/2)$の5次の近似式を次に示します．

$$\sin\left(\frac{\pi}{2}x\right) \cong a_1 x + a_3 x^3 + a_5 x^5, \quad -1 \le x \le 1 \quad \cdots\cdots (G.5)$$

$$a_1 = 1.57065736$$
$$a_3 = -0.64345777$$
$$a_5 = 0.07293465$$

本書付属CD-ROMの…¥BcppB¥Sin_ChebInt¥のフォルダには，区間$[-1, 1]$における$\sin(\pi x/2)$の第1近似式をチェビシェフ補間で求めるために，C++言語で作成したプログラムの例を収録しています．

図G.1には，チェビシェフ補間で求めた近似式である式(G.5)と，マクローリン展開で求めた近似式の絶対誤差のようすを示します．この図からわかるように，誤差の絶対値の最大値で比較すると，同じ次数ではチェビシェフ補間で求めた近似式の方が誤差は小さいことがわかります．誤差の絶対値の最大値がチェビシェフ補間で求めた5次の近似式よりも小さくなるのは，マクローリン展開の9次の項まで使った場合です．



図G.1
$\sin(\pi x/2)$について，チェビシェフ補間で求めた近似式の誤差とマクローリン展開で求めた近似式の誤差