連載第4回

数値計算ソフトウェアを用いたアルゴリズム検証

関数計算アルゴリズムの検証

河原崎浩也

CD-ROM 関連データ

関数計算でよく使われるアルゴリズムから，CORDIC法と多項式による近似の二つの方法をとりあげ，計算過程の概略をシミュレーションします．計算の収束過程や誤差分布をグラフ化し，アルゴリズムの特徴を明らかにします．

CORDIC法による三角関数の計算

●CORDICアルゴリズムの原理

CORDICアルゴリズム（Coordinate Rotation Digital Computer）は，関数値を複素平面上のベクトルとしてとらえ，複素数の演算によってベクトルを回転させて真値に反復収束させるものです（**図1**）．初期値と収束条件の定め方により，sin，cos，$\tan^{-1}$，sinh，cosh，exp，logなどの関数値を同一のアルゴリズムで求めることができます．このアルゴリズムは比較的小規模な回路で実現できるため，関数電卓で使用されています．以下では，sinとcosに話を限って，アルゴリズムの概要を説明します．

複素平面上で，ある角度θをもつ原点を起点とする単位ベクトル$\boldsymbol{f}$を考えます．実数部をx，虚数部をyとすると，x，yは三角関数に対応します．

$$\begin{cases} x = \cos\theta \\ y = \sin\theta \end{cases} \quad \begin{cases} x^2 + y^2 = 1 \\ \theta = \tan^{-1}\dfrac{y}{x} \end{cases}$$

ここで，初期状態として θ＝0，x＝1，y＝0とします．求める関数の引き数として目標位相角tが与えられたとして，ベクトル$\boldsymbol{f}$を回転させてθ＝0→tにすることを考えます．ここで，複素平面上でベクトルに角度αの回転をさせる演算子を定義し，補正ベクトル$\boldsymbol{g}$と呼ぶことにします．この変換を行列形式で次のように書き表すことができます．

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \begin{pmatrix} 1 & p\delta \\ -p\delta & 1 \end{pmatrix} \begin{pmatrix} x_n \\ y_n \end{pmatrix} = \boldsymbol{g}\cdot\boldsymbol{f}$$

パラメータpは，1または−1で，変換の方向（回転の方向）を決めます．この変換でベクトル$\boldsymbol{f}$は，絶対値が，$\sqrt{1+\delta^2}$ 倍されます．ここで，パラメータδを以下のように定めます．

$\delta = 2^{-(n-1)} = 1, 1/2, 1/4, 1/8\cdots$

位相角の変化は，

$\theta_{n+1} = \theta_n + \alpha$　ただし，$\alpha = \tan^{-1}(\delta)$

結局，この変換は，長さが $\sqrt{1+\delta^2}$ で，位相角がαのベクトル$\boldsymbol{g}$を掛けることと等価です．位相角度補正方向の符号pは，θ−tの現在の符号をみて，θ−tと反対の符号にとります．このベクトル回転（補正ベクトル$\boldsymbol{g}$の乗算）を，誤差（θ−t）が十分に小さくなるまで繰り返します（実際は，θの初期値をtとし，これがゼロになるようにpの符号を決める）．

関数ベクトル$\boldsymbol{f}$の絶対値は，1から出発して，最終的に，

$$|\boldsymbol{f}| = \sqrt{1+\left(\frac{1}{1}\right)^2} \times \sqrt{1+\left(\frac{1}{2}\right)^2} \times \sqrt{1+\left(\frac{1}{4}\right)^2} \times \cdots \times \sqrt{1+\left(\frac{1}{2^{n-1}}\right)^2} \approx 1.646760258\cdots$$

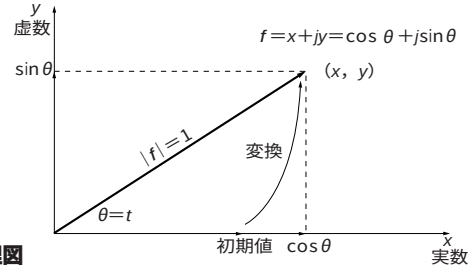
となり，tに無関係に一定になります．そこで，$\boldsymbol{f}$の初期値の絶対値を，この逆数，

$$|\boldsymbol{f}| = \frac{1}{0.60725\cdots} \qquad \boldsymbol{f} = \frac{1}{0.60725\cdots} + j0$$

とし，初期位相θ＝0とすると，最後の収束結果は$\boldsymbol{f}$の絶対値＝1，θ−t＝0となり，$\boldsymbol{f}$の実数部＝cos t，虚数部＝sin tとなって，三角関数の値が得られたことになります．

●Excelによる計算

x，yの計算部分は，x，yに係数$1/2^n$を掛けて加減算するだけなので，下方ビット・シフトと，加減算回路で実現できます．$\theta_{n+1} = \theta_{n\pm\alpha}$の計算にtanの逆関数値を加減計算するので，これをあらかじめ計算しておいてテーブルとしてもっている必要があります．また，目標位相角tとして与えることができる引き数の範囲があり，$\sum_{n=0}^{\infty}\tan^{-1}\left(\pm 2^{-n}\right)$ の収束範囲から，$|t| < 1.74\cdots$



〔図1〕CORDICの原理図

[radian]に限られます．補正するベクトル$\boldsymbol{g}$の大きさが決まっているために，収束の過程はなめらかではなく，目標を飛び越えたり戻ったりという動きになります．付属CD-ROMに収録したワークシートCordic.xls(**図2**)では，収束の過程を見られるように，あるtにおける$\boldsymbol{f}$の変化を複素平面上のベクトル軌跡としてグラフ化しています．結果の精度と繰り返し回数の関係では，おおざっぱに言って，1回の繰り返しで誤差の最大値が約半分になるので，2進数nビットの精度を求めるときはおよそn回の反復収束が必要になります(**図2**(c)，(d)を参照)．

ワークシートでは，B列を引き数位相角t，C，D列を三角関数真値x，yとして，シートの右方へ進むごとに，繰り返し1回目，繰り返し2回目…と続いています．このシートはVBマクロは使用していません．

ワークシート上の計算過程における，繰り返し1回目から6回目までの，sin関数値の収束のようすを表すグラフが**図2**(b)です．補正ベクトルがしだいに小さくなりながら矩形波の形で加算されて，関数値が2進法的に収束していくようすを見て取ることができます(Mathematicaの例はCORDIC.PDF)．

多項式による関数の近似

関数計算法のもう一つの代表例として，多項式近似を検討してみます．近似多項式の作り方はいろいろありますが，チェビシェフ多項式補間による関数近似法をとりあげます．ここでは，簡単にチェビシェフ近似と呼ぶことにします．

余談ですが，x86系CPUの関数計算は，486まではCORDICでしたが，Pentiumから多項式近似になったという話を聞いたことがあります．CORDICは，実現するための回路は加減算とシフトだけで簡単ですが，必要な精度を得るために多くの反復回数を要します．多項式近似は乗算回路が必要で比較的大規模になりますが，計算ステップ数は近似式の次数で決まり，実行速度上，有利です．Pentiumの場合，単純な多項式近似ではなくて，特殊なアルゴリズム上の工夫が行われているとのことです．

●関数の近似式を求めることの一般論

近似される関数として，近似する式を，

$$g(x) = a_0T_0(x) + a_1T_1(x) + a_2T_2(x) + \cdots + a_nT_n(x)$$

とします．ここで，a_0，a_1，a_2,…，a_nはある係数，$T_0(x)$，$T_1(x)$，$T_2(x)$，…，$T_n(x)$はある既知の関数です．つまり，ある基底関数$T_0(x)$，$T_1(x)$，$T_2(x)$，…，$T_n(x)$が存在し，$f(x)$をそれらの1次結合として合成しようというわけです．$f(x)$と$T_0(x)$，$T_1(x)$，$T_2(x)$，…，$T_n(x)$が与えられたとき，係数a_0，a_1，a_2,…，a_nを求めるというのが，「近似式を求める」ということの数学的な意味になります．

基底，および1次結合とは，線形代数で使用されている基礎概念です．ここでは，多項式空間を，ベクトル空間としてとらえています．基底関数$T_0(x)$，$T_1(x)$，$T_2(x)$の組は線形独立である必要があり，かつ直交性の高いものが良いとされます．チェビシェフ多項式を関数ベクトルとして見たとき，区間(−1，1)では，ある条件で，異なる次数の式の間の内積がゼロ(＝直交)になるという性質があり，上記の要求を満たします．

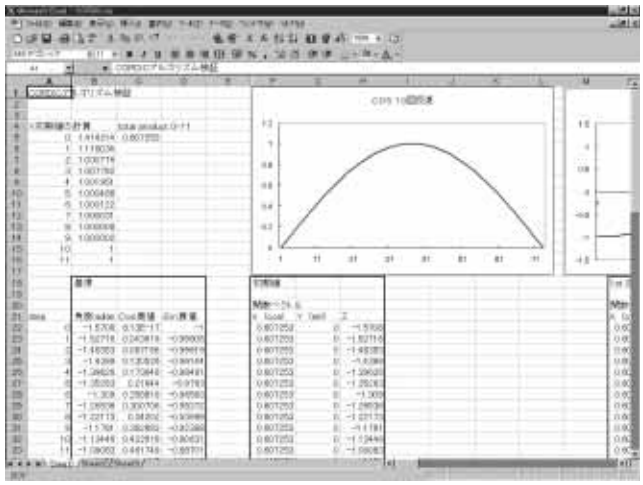
●ワイエルシュトラスの多項式近似定理

多項式によって任意の精度の近似式を求められることの原理は，ワイエルシュトラスの近似定理によって与えられます．

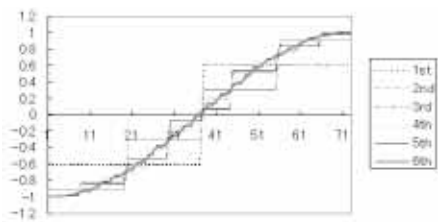
［定理］xを，実数体上の有界な閉区間$[a, b]$上の実数値連続関数に属する任意の数とする．任意に与えられた正数εに対して有理数p_iを係数とする多項式，

$$P(t) = p_0 + p_1t + p_2t_2 + \cdots + p_nt_n$$

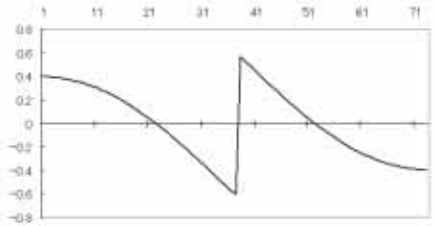
が存在して，$\| x - P \|_\infty < \varepsilon$をみたす(ただし，$\| x \|_\infty$は関数ベクトルの最大値ノルムを表す)．



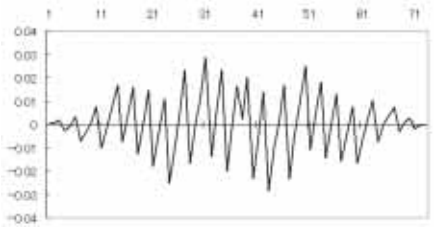
(a) **Excelの画面**



(b) **sin関数計算の収束過程**



(c) **sin誤差　1回目**



(d) **sin誤差　6回目**

〔図2〕Cordic.xls