複素円板法による初等関数の正則性判定システムの構築

2012SE093 加藤　里菜
指導教員：杉浦洋

1　はじめに

現代のコンピュータにおいて，高速な数値計算ができるのは，実数を浮動小数点数で近似して計算を行うからである．しかし，近似であるため厳密に正しいことが保証されていない．そこで，計算結果がどのくらい正しいか検算することが重要になってくる．これを精度保証付き数値計算という．

2　円板解析の基本的な定義

2.1　円板の定義

中心 $c \in \mathbb{C}$，半径 $r \in \mathbb{R}$ の複素閉円板領域を

$$Z = \langle c; r\rangle = \{z : |z - c| \le r\} \tag{1}$$

と表し，以後これを単に円板と呼ぶ．円板の中心は $c = mid\ Z$，半径は $r = rad\ Z$ と表す．また，円板全体の集合を $\mathbb{KC} = \{\langle c; r\rangle : c \in \mathbb{C}, r \in \mathbb{R}, r \ge 0\}$ と書く．

2.2　円板四則演算の定義

円板四則演算を定義する．2 つの円板 $A = \langle a; r\rangle$, $B = \langle b; s\rangle \in \mathbb{KC}$ の四則演算を次のように定義する．

加減算

$$A \pm B \equiv \langle a \pm b; r + s\rangle = \{z \pm w : z \in A, w \in B\},$$

逆円板

$$\frac{1}{A} \equiv \left\langle \frac{\bar{a}}{|a|^2 - r^2}; \frac{r}{|a|^2 - r^2} \right\rangle = \left\{ \frac{1}{z} : z \in A \right\} \quad (0 \notin A),$$

乗算

$$AB \equiv \langle ab; |a|s + |b|r + rs\rangle \supseteq \{zw : z \in A, w \in B\},$$

除算は逆円板を用いて

$$A/B \equiv A(1/B) \supseteq \left\{ \frac{z}{w} : z \in A, w \in B \right\}.$$

3　正則性判定システム

3.1　正則判定システム

第 2 章で説明した方法で正則判定システムの機能をもつ円板四則演算 AddD, SubD, MltD, DivD 逆数関数 RspD を以下のようにな定義に従って作成した．円板の中心と半径は，Mathmatica の区間演算機能を用いて計算し，丸め誤差を押さえ込み，円板の包含性を理論的に保証した．

・$\mathrm{AddD}[A, B] = A + B$

・$\mathrm{SubD}[A, B] = A - B$

・$\mathrm{MltD}[A, B] = A \times B$

・$\mathrm{DivD}[A, B] = A \div B$

・$\mathrm{RspD}[A] = 1/A$

また, 第 3 章の方法に基づいて指数関数, 対数関数（主値）に対する

・$\mathrm{ExpD}[A] = Exp(A)$

・$\mathrm{LogD}[A] = Log(A)$

を作成して，2 つの関数を用いて三角関数の円板関数

$$\mathrm{SinD}[A] = \frac{\mathrm{ExpD}[iA] - \mathrm{ExpD}[-iA]}{2i}$$

$$\mathrm{CosD}[A] = \frac{\mathrm{ExpD}[iA] + Exp[-iA]}{2}$$

$$\mathrm{TanD}[A] = \frac{\mathrm{SinD}[A]}{\mathrm{CosD}[A]}$$

を作成した.

3.2　正則性フラグ

正則フラグ Rflag とは，基本関数が入力円板で正則かどうか知らせるものである．これは円板のデータに付加する．入力円板で Rflag $= True$ なら，入力円板が関数の特異点を含まないなら，Rflag $= True$，含むなら Rflag $= False$ を出力円板に付加する．入力円板で Rflag $= False$ ときは，無条件に出力円板に Rflag $= False$ を付加する．そして，基本関数の合成関数として表される初等関数においても，入力円板の Rflag $= True$ として関数計算し，その出力円板の正則フラグを見ることにより，入力円板で関数が正則かどうか判定できる．

基本関数の合成関数 f において，出力円板の Rflag が $True$ なら，入力円板において f は必ず正則である.「f は入力円板で正則である」は，数学的に真の命題である．出力円板の Rflag が $False$ でも，入力円板において f は正則である可能性はある．合成関数の計算の途中結果において要素となる関数の値域は，円板で包囲され大きめに扱われるからである．

3.3　正則領域の確定

領域を円板で覆い尽くし，それらの円板で関数が正則かどうか判定することで，関数が正則となる部分領域を確定することが出来る．まず，大きな円板で正則性を判定する．非正則と判定された領域は，小さな円板により判定を精密化する．

［例 1］[-5,5]*[-5,5] で

$$f(z) = \frac{\sin z}{1 - e^{2z}}$$

の正則領域を求める．円板を小さくしてゆくと，関数の極 $z = 0,\ \pi i,\ -\pi i$ が白抜きで見えてくる．

例 1 では，円板を小さくして操作を行う方法は手動で行った．それを，プログラムで自動的に行う再帰型関数 $\mathrm{Reg}[f, A, rmin]$ を作成した．円板 $A = \langle c; r\rangle$ の内接正六角形内部で，関数 f の正則部分領域を正則円板の和集合として表す．正則性判定に用いる円板の最小半径を $rmin$ とする．

最初，円板 A で f の正則性を判定する．正則なら，円板 A のみを要素とする集合 $\{A\}$ の内接小六角形を出力する．すなわち，$\mathrm{Reg}[f, A, rmin] = \{A\}$ である．

A で非正則と判定されたら，図 1 のように A の内接六角形を覆う 6 個の小円板 $A_1,\ A_2,\ A_3,\ A_4,\ A_5,\ A_6$ で同じ操作を行い，小円板からの出力円板集合の和集合を最終出力とする．すなわち

$$\mathrm{Reg}[f, A, rmin] = \bigcup_{i=1}^{6} \mathrm{Reg}[f, A_i, rmin]$$

領域細分は，円板の半径が $rmin$ 以下になれば停止し，空集合を出力する．

[例 2] 関数

$$f(z) = \sin \frac{z}{1 - e^{2z}}$$

について $A = \langle 0.01, 1\rangle, rmin = 0.01$ で実験した結果を図 2 に示す．内接正六角形内部の白抜きの部分が正則円の和集合である．孤立特異点 0 が黒い点として表れる．正則円板の個数は 166 個であった．

[例 3] 関数

$$f(z) = \frac{1}{\sin z}$$

について $A = \langle 0.01, 4\rangle, rmin = 0.01$ で実験した結果を図 3 に示す．内接正六角形内部の白抜きの部分が正則円の和集合である．孤立特異点 0, $\pm\pi$ が黒い点として表れる．正則円板の個数は 457 個であった．

ExpD は Taylor 円板で設計されているが，SinD は ExpD の合成関数であるために出力円板が大きめである．これが，例 2 と例 3 の正則円板数の差に表れていると思われる．

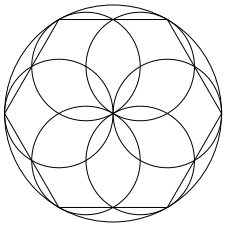

図 1

4　おわりに

本研究では四則演算，指数関数，対数関数，および三角関数の円板関数を作成した．指数関数，対数関数に関しては，Taylor 円板によるくるみ関数を用いた．三角関数は，指数関数の合成関数として，くるみ関数を構成した．くるみ関数の出力円板の中心と半径は，Mathmatica の区間演算で計算し，包含性を保証した．

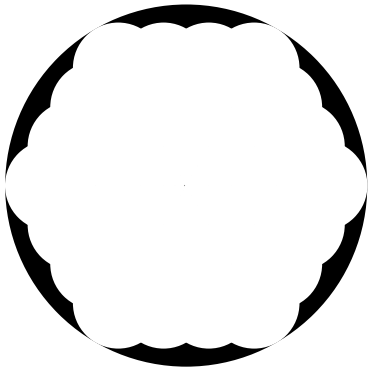

図 2

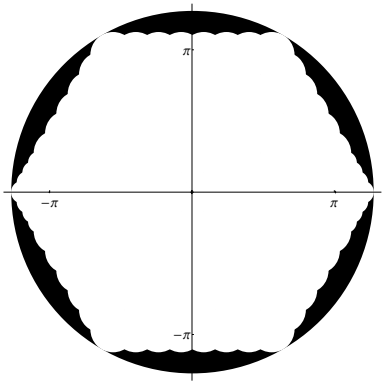

図 3

くるみ関数の出力に正則性フラグを付加した．これにより，上記基本関数が入力円板で正則かどうかを知らせる．上記基本関数の合成関数として表される初等関数においても，入力円板で正則かどうかは出力円板の正則フラグを見ることにより，判定できる．

今後の課題として，使いやすいシステムの開発があげられる．現在のシステムは，Mathmatica 上で実現されているために，二項演算まで，関数として書かねばならず，使用が煩雑である．例えば，関数 $f(z) = \frac{sinz}{1-exp(2z)}$ はこのシステムでは $ff[c] := \mathrm{MltD}[\mathrm{SinD}[c], \mathrm{RspD}[\mathrm{SubD}[one, \mathrm{Exp}[\mathrm{MltD}[two, c]]]]];$ と書かねばならない．これは，C++などオブジェクト指向言語の関数のオーバーロード機能を使えば，圧倒的に改善される．

5　参考文献

[1] 斉藤裕樹：精度保証付き数値積分，南山大学数理情報学部情報システム数理学科 2009 年度卒業論文，2010 年度卒業論文，2011 年度卒業論文

[2] 森正武:数値解析と複素関数論，筑摩書房，東京，1975，pp.216-218

[3] 斉藤裕樹：精度保証付き数値積分

[4] 岸正倫，藤本担孝：複素関数論